

Overlapping Grids for Solving PDEs on Complex Moving Geometry



Bill Henshaw

Centre for Applied Scientific Computing,
Lawrence Livermore National Laboratory,
Livermore, CA, USA 94551

[www.llnl.gov/casc/Overture](http://llnl.gov/casc/Overture)

Numerical Grid Generation in Computational Field Simulation
9th International Conference on
San Jose, California June 2005.

- Introduction to overlapping grids and the Overture framework
- Overview of grid generation capabilities
- Incompressible Navier-Stokes equations
- Reactive Euler equations
- Multigrid on overlapping grids
- Adaptive mesh refinement
- Moving grids
- Parallel paradigm in Overture
- Parallel results (preliminary)
- Movies

Outline

Acknowledgments

Supported by

Department of Energy, Office of Science

MICs Program: Mathematical, Information, and Computational Sciences

Current Overture developers

Kyle Chand (hybrid mesh algorithms and hybrid mesh generation)

Bill Henshaw

Collaborators

Petri Fast (LLNL) (deforming boundaries)
Don Schwendeman (RPI) (reactive-Euler equations, multiphase)

Overview: Overture is a collection of C++ classes that can be used to solve partial differential equations on overlapping and hybrid grids.

Key features:

- provides a high level interface for rapid prototyping of PDE solvers.
- built upon optimized C and fortran kernels.
- support for overlapping grids and hybrid grids.
- provides a library of finite-difference operators: conservative and non-conservative, 2nd, 4th, 6th and 8th order accurate approximations.
- support for moving grids and block structured adaptive mesh refinement
- extensive grid generation capabilities (overlapping and hybrid)
- CAD fixup tools
- extensible data-base support for saving grids, solutions etc.
- graphics (interactive and post-processing)
- support for parallel computations

new approaches.

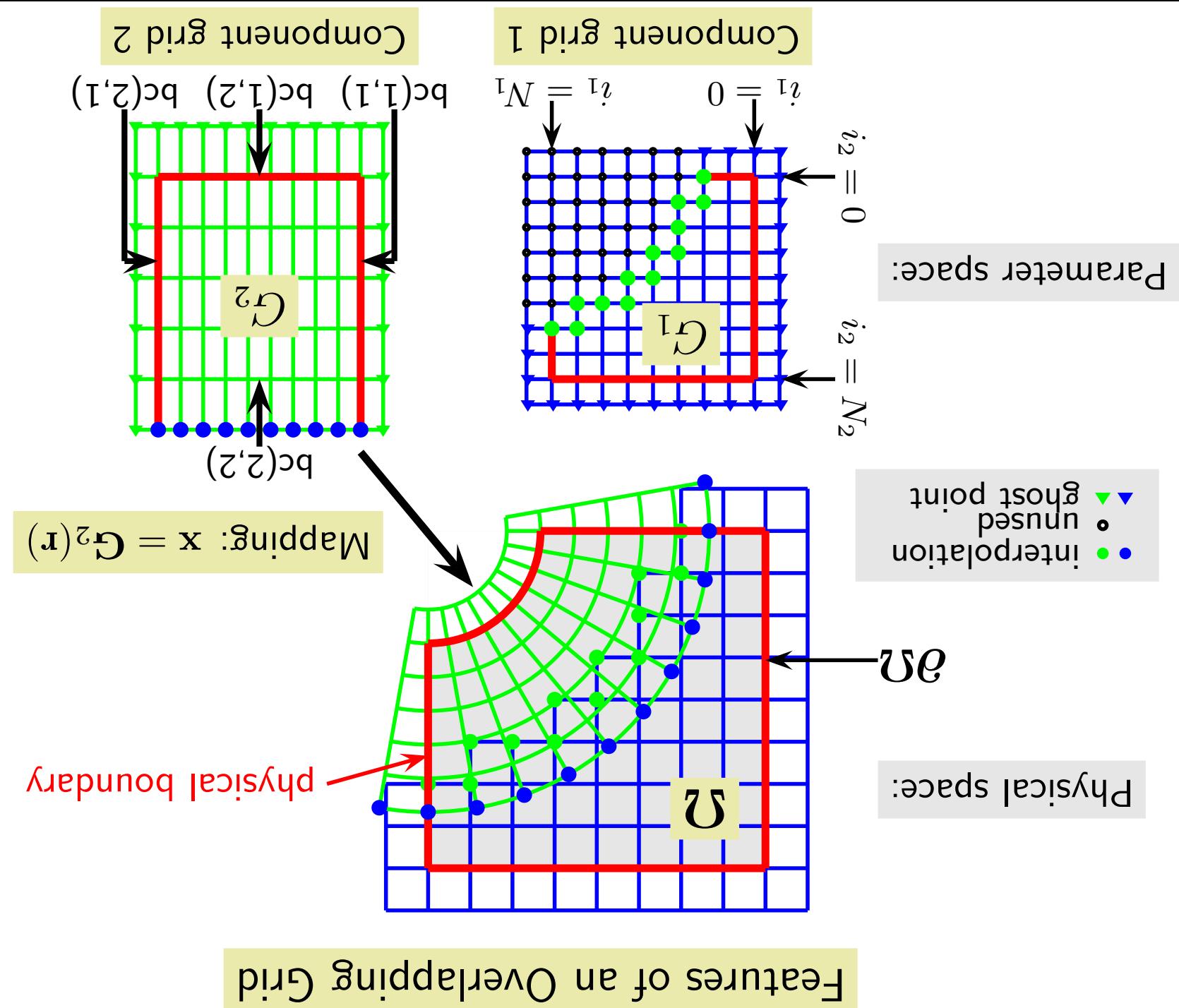
At the same time it is very useful to have high-level interfaces for prototyping.

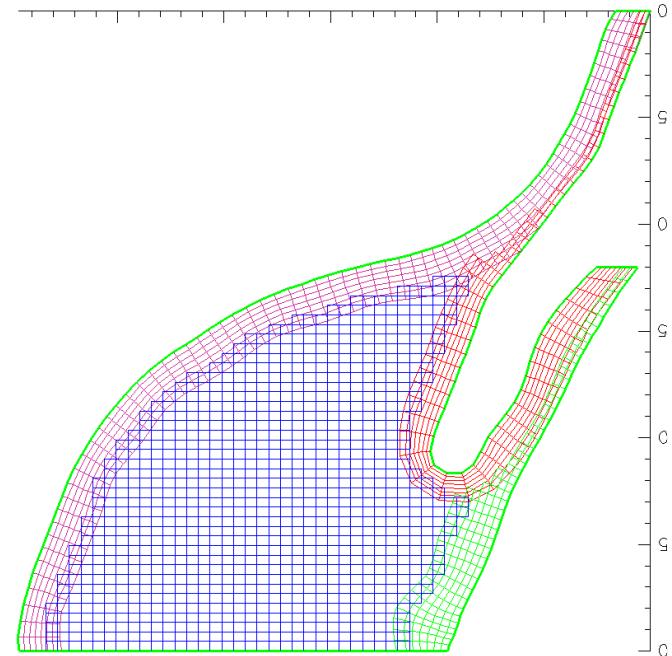
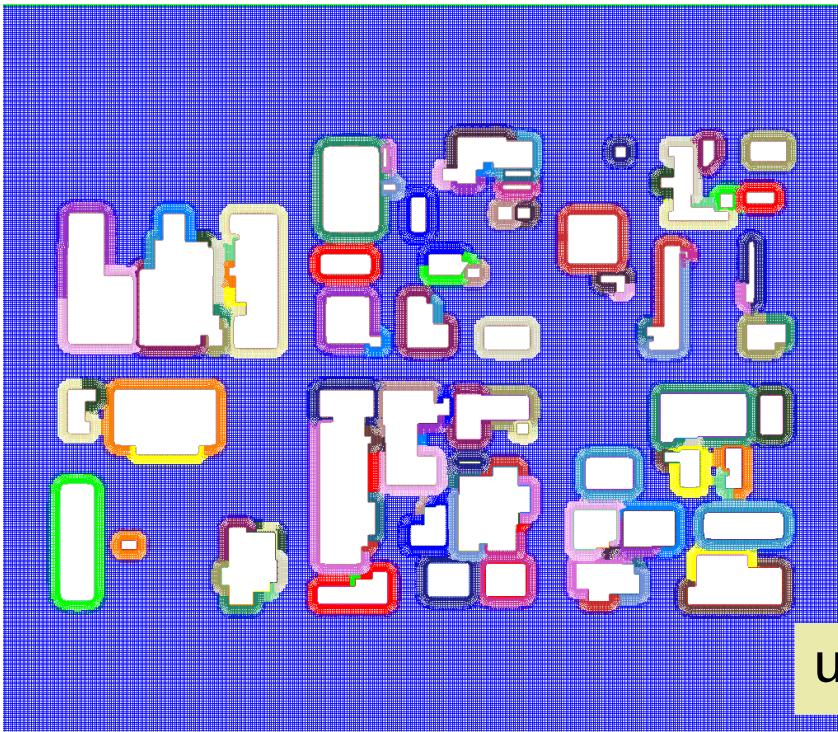
Our goal is to be as fast as codes written for Cartesian Grids.

It is critical that solvers can be made as fast and efficient as possible.

- PDE solvers built upon Overture include:
 - Ogmf: multigrid solver for elliptic boundary value problems
 - ◆ can achieve near text-book convergence rates and efficiency.
 - OverBlown: solver for the incompressible Navier-Stokes,
 - ◆ OverBlown: solver for the incompressible Navier-Stokes, compressible Navier-Stokes, and reactive Euler equations.
 - MX: time domain Maxwell's equations solver:
 - ◆ hybrid-grid, second-order stabilized DSI scheme.
 - ◆ overlapping-grid, fourth-order accurate, parallel.

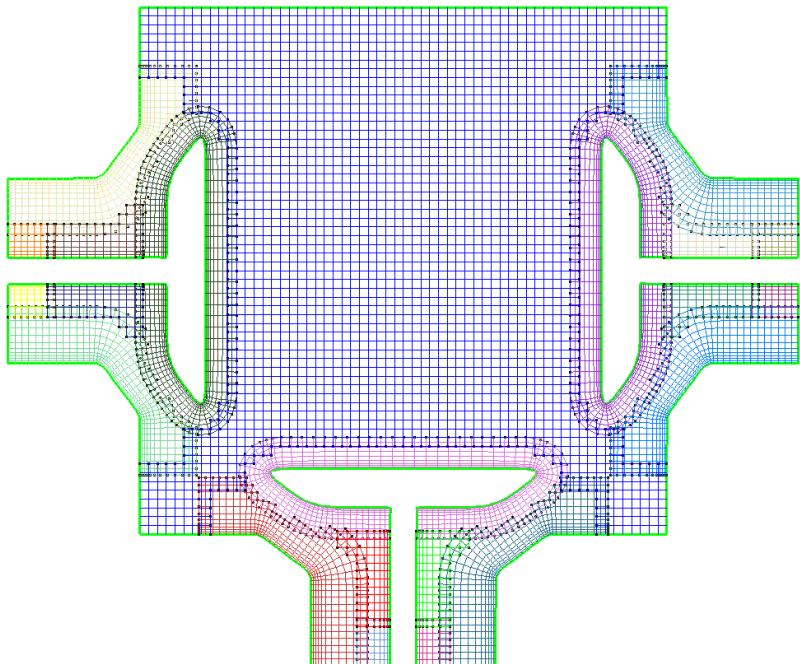
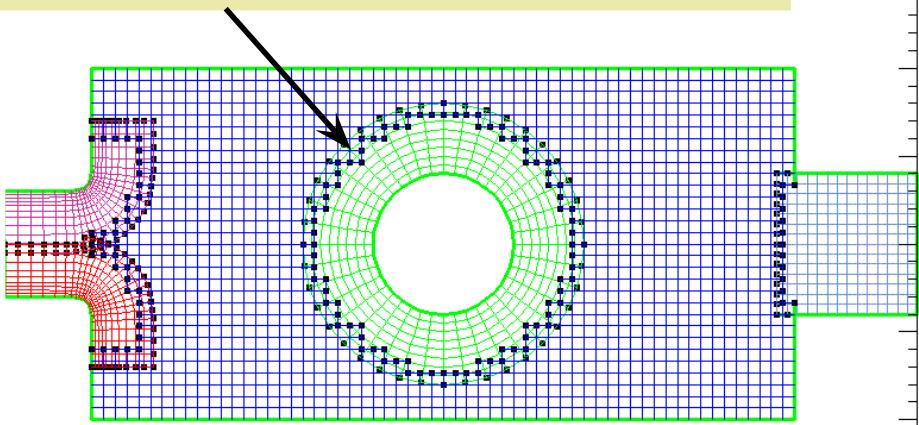
Overture Overview continued...

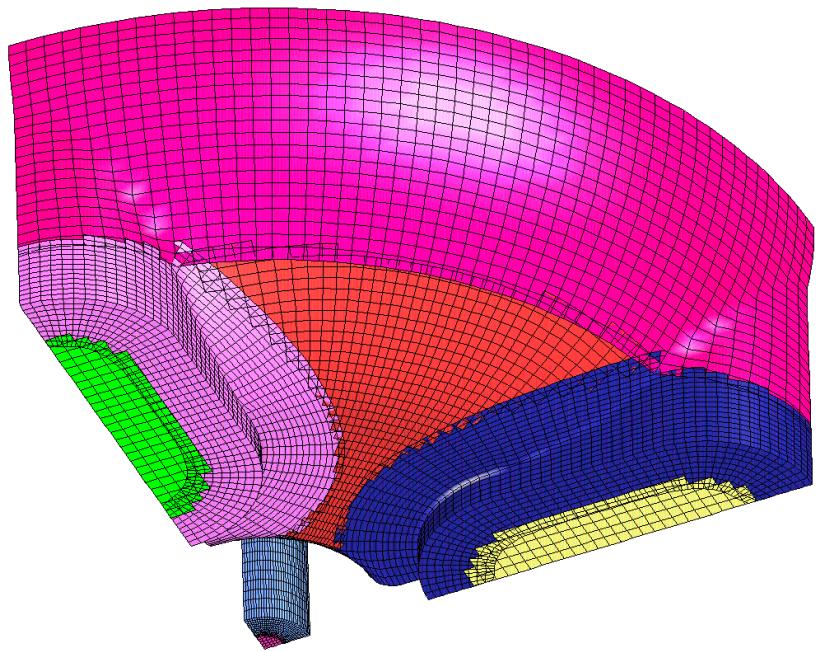
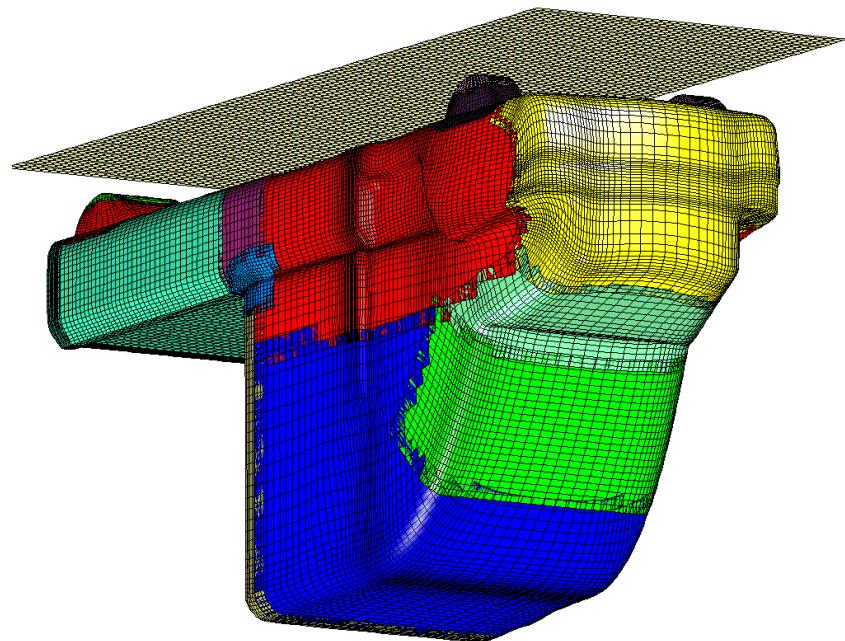




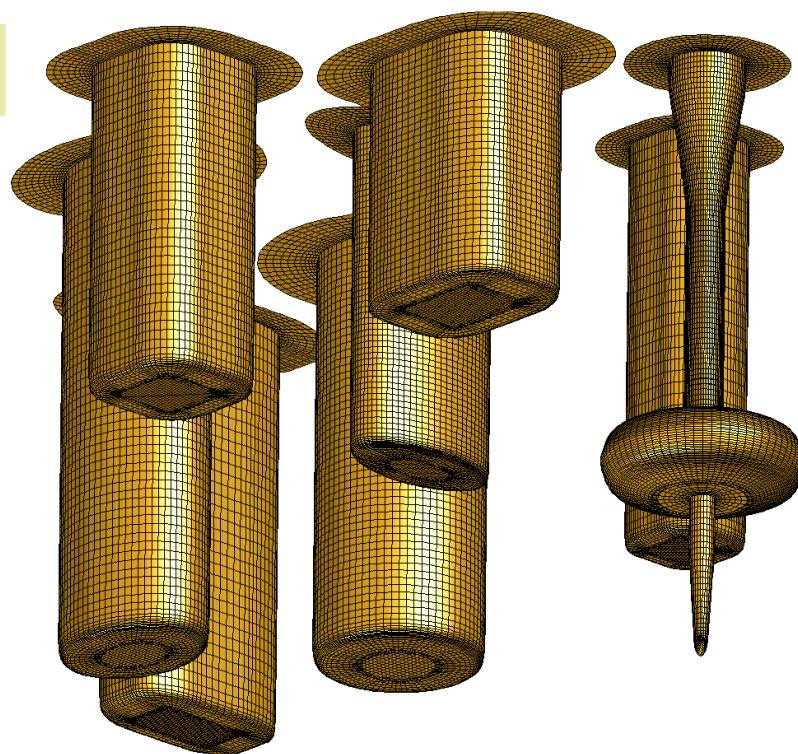
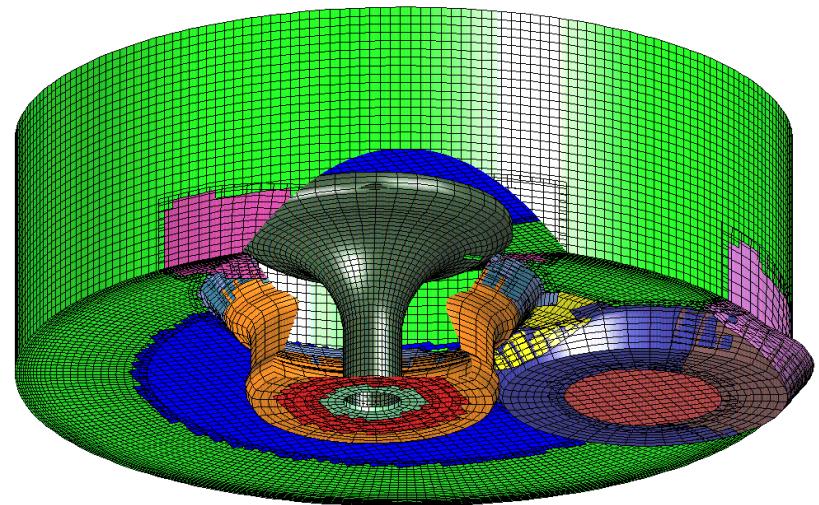
2D overlapping grids built with Qgen

Solutions coupled by interpolation

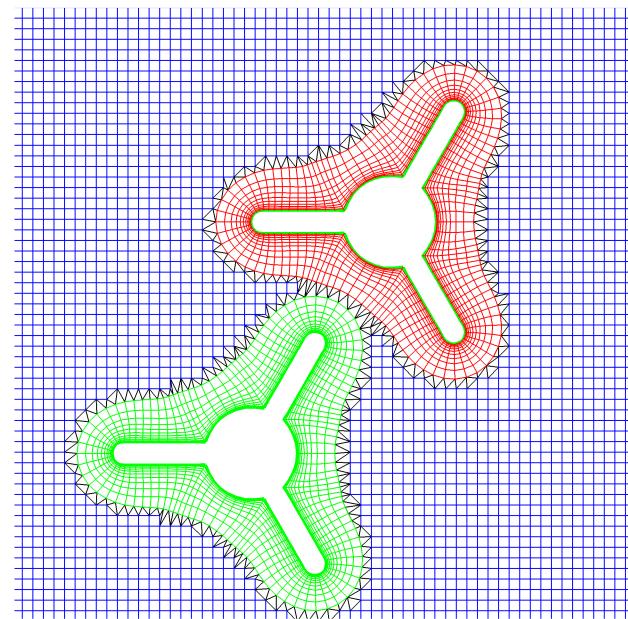
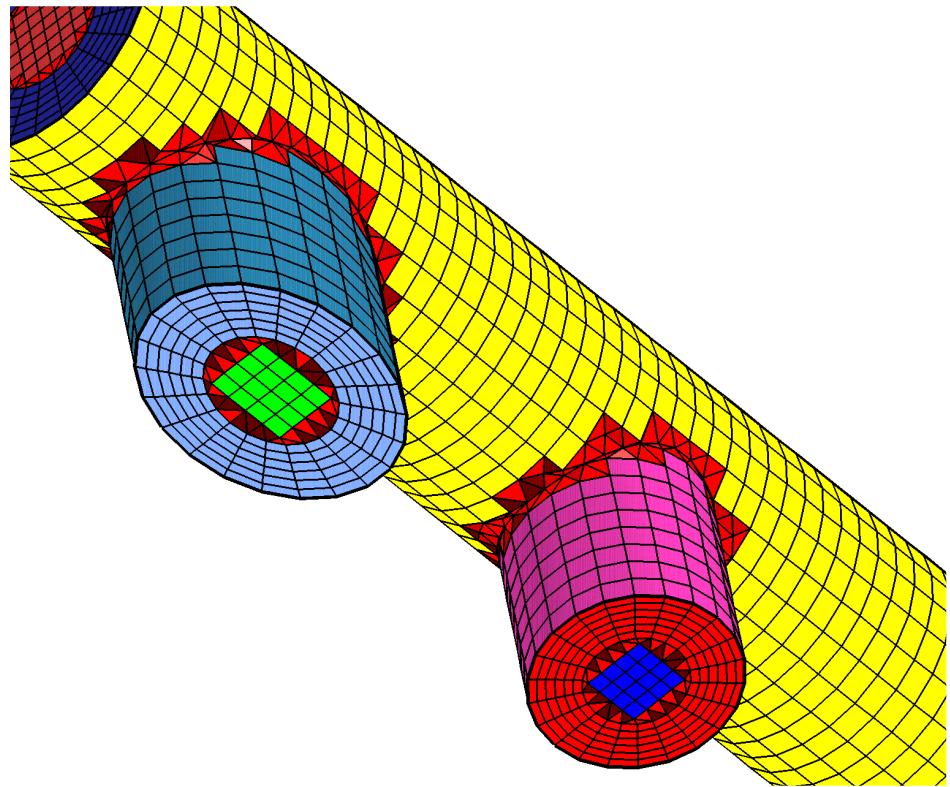
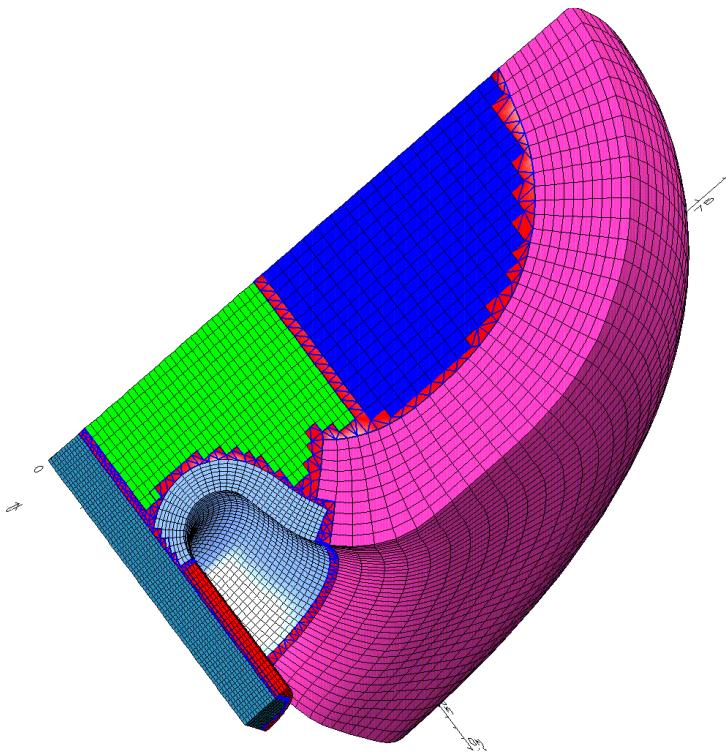




3D overlapping grids built with OpenFOAM



Hybrid grids built with Overture



Boxlib

PETSc

HDF
OpenGL

A++/P++

Graphics

mbuilder
rap, htype

Grid Generation
CAD fixup

Mappings

Operators

GridFunctions

Grids

AMR

Unstructured
Ugen

Overlapping
Ugen

Multi-grid
Omg

Linear Solvers
Oges

INS, CNS
OverBlown

Overture

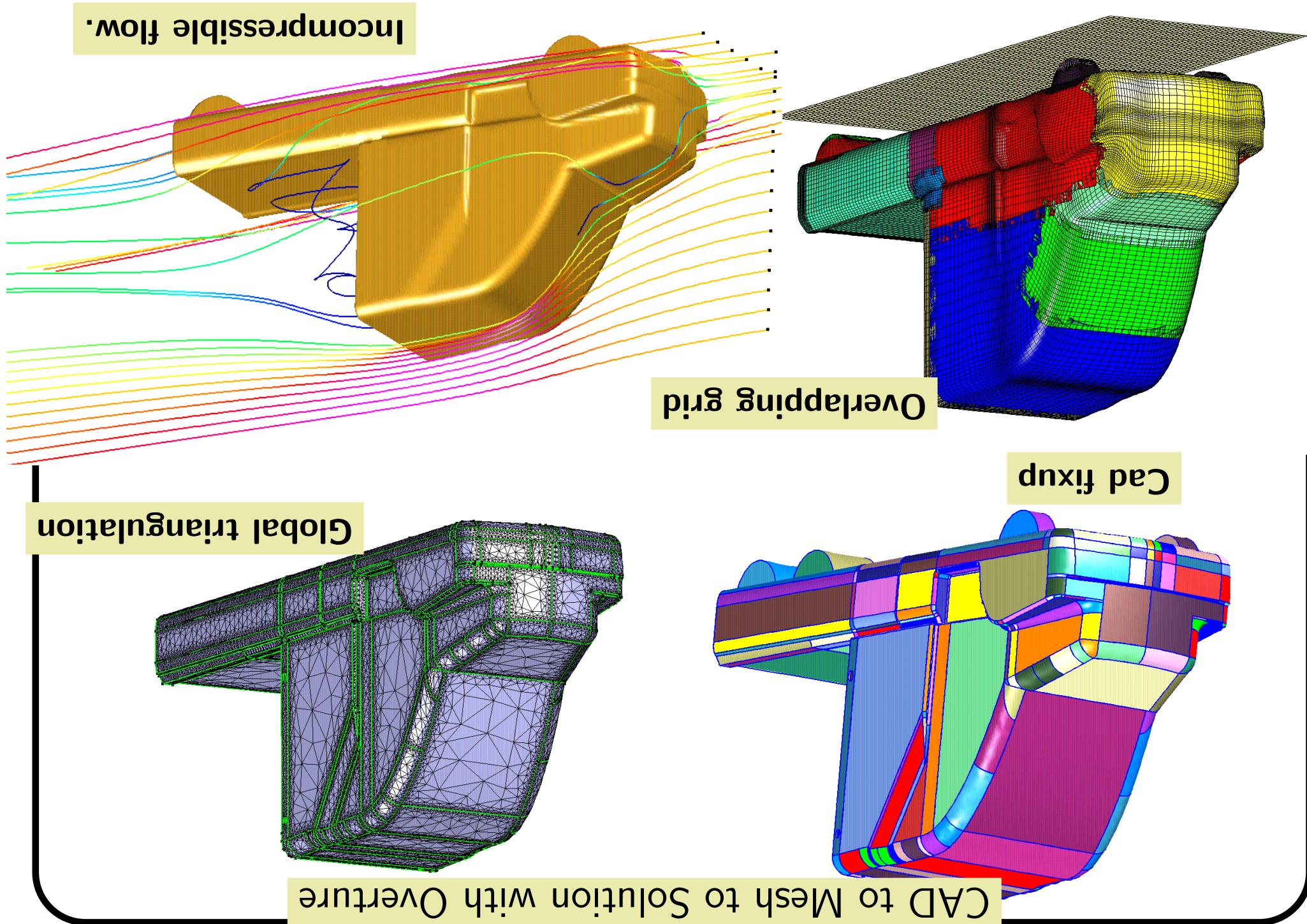
```

Solve  $u_t + au_x + bu_y = \nu(u_{xx} + u_{yy})$ 

CompositeGrid cg; // create a composite grid
getFromDatabaseFile(cg, "myGrid.hdf");
floatCompositeGridFunction u(cg); // create a grid function
floatCompositeGridOperators op(cg); // operators
u.setOperators(op);
CompositeGridOperators op(cg); // operators
float t=1.; // time
float a=1., b=1., nu=1;
float dt=.005, step=0, step<100; step++
for (int step=0; step<100; step++)
{
    float dt*( -a*u.x() -b*u.y() +nu*(u.xx() +u.yy()) );
    // forward Euler
    t+=dt;
    u.interpolate();
    u.applyBoundaryConditions();
}

```

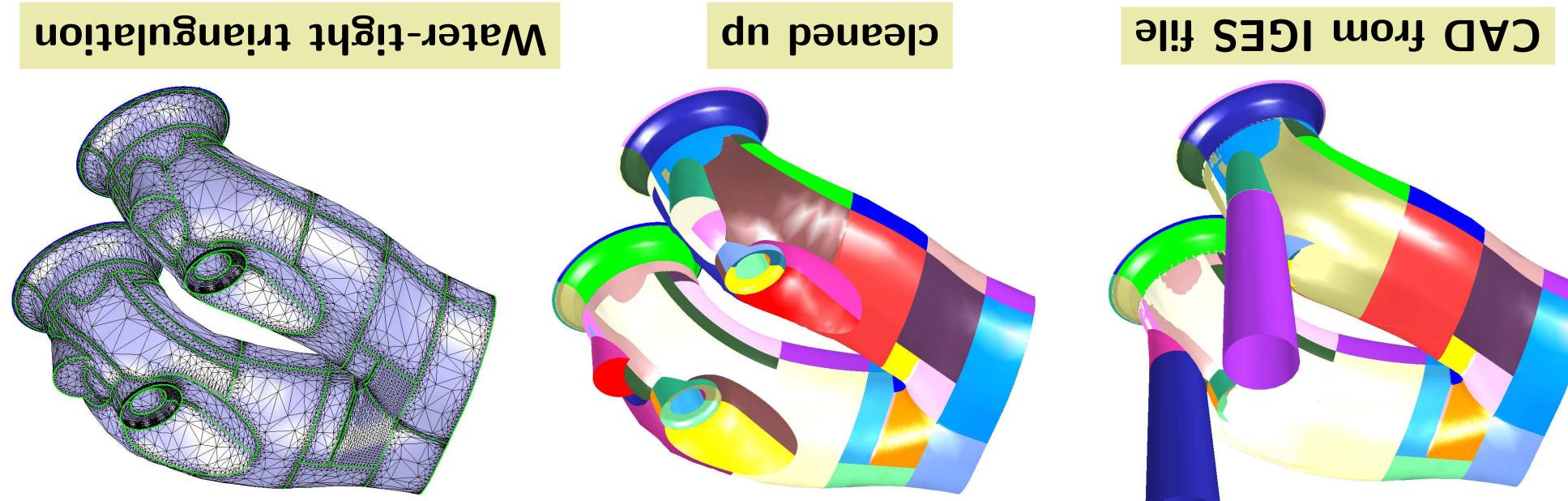
Overture supports a high-level C++ interface (but is built mainly upon Fortran kernels):



Summary of the Grid Generation Capabilities in Overview

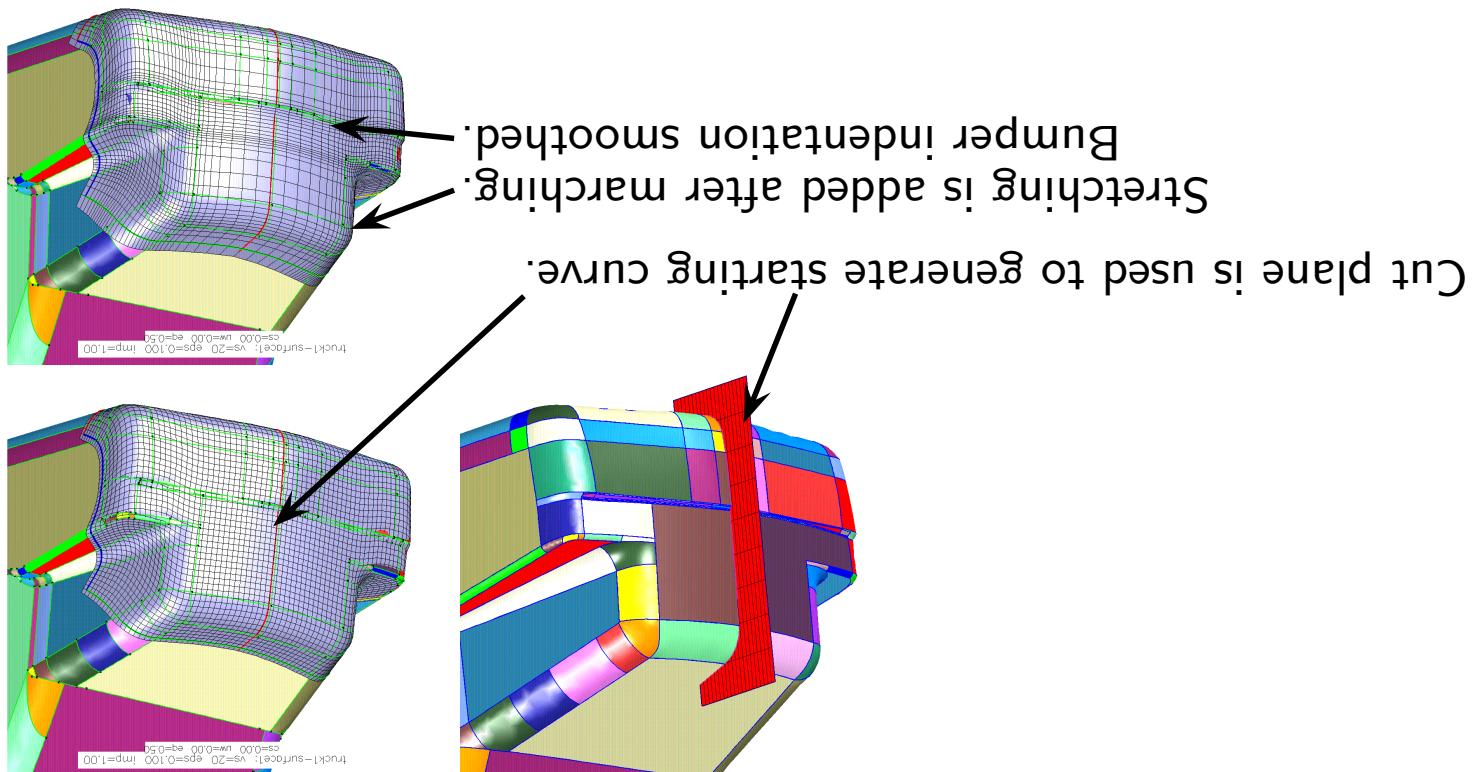
Component Grid Generation:

- Analytic mappings: square, annulus, box, cylinder, sphere...
- Splines, NURBS, transfinite interpolation
- rotation, scaling, translation, bodies of revolution
- Grid stretching
- elliptic grid generation and smoothing
- hyperbolic grid generation



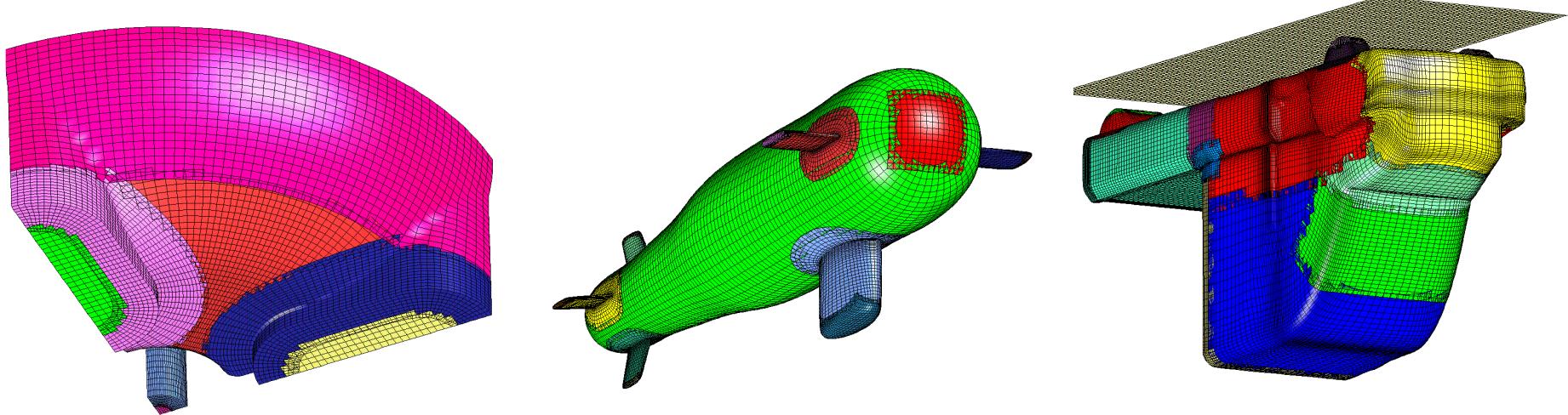
- Goal: tools for fixing and editing CAD and building water-tight representations.
- Key features:
- IGES reader which detects errors in CAD,
 - interactive editing of CAD representation,
 - interactive editing of trim curves,
 - nearly automatic generation of a water-tight triangulation,
 - fast algorithms for projecting points onto the surface.

CAD fixup, editing and topology generation



- Goal:** rapid generation of boundary conforming grids for CAD geometries.
- Key features:**
- ◊ generates surface and volume grids on CAD geometries,
 - ◊ flexible choice of starting curves and boundary conditions,
 - ◊ robust treatment of corners and edges,
 - ◊ smoothing and stretching of grid lines, smoothing of geometry.

Hype: hyperbolic grid generator



- Goal: a fast, flexible and robust grid generator for overlapping grids.
- Key features:
- ◊ general and automatic algorithm for determining connectivity information,
 - ◊ robust hole cutting algorithm,
 - ◊ extensive diagnostics,
 - ◊ robust treatment of shared boundaries, C-grids and H-grids
 - ◊ optimized algorithms for moving grids and AMR
 - ◊ support for high-order discretization and interpolation

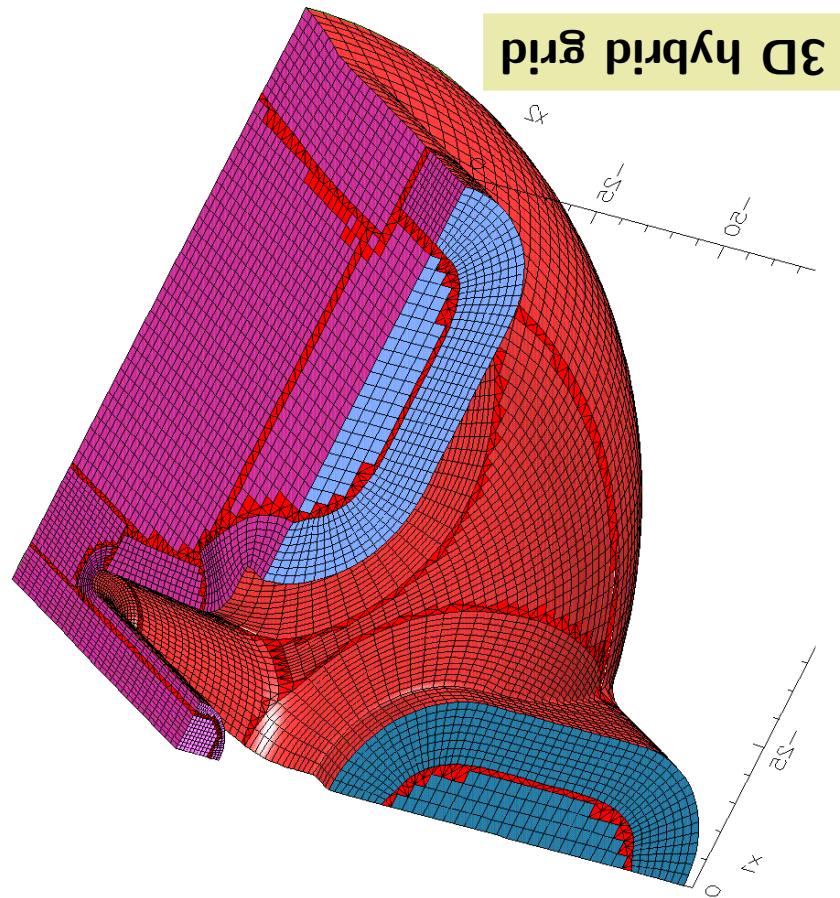
Ogen: overlapping grid generator

Ugen: hybrid grid generation (Kyle Chand)

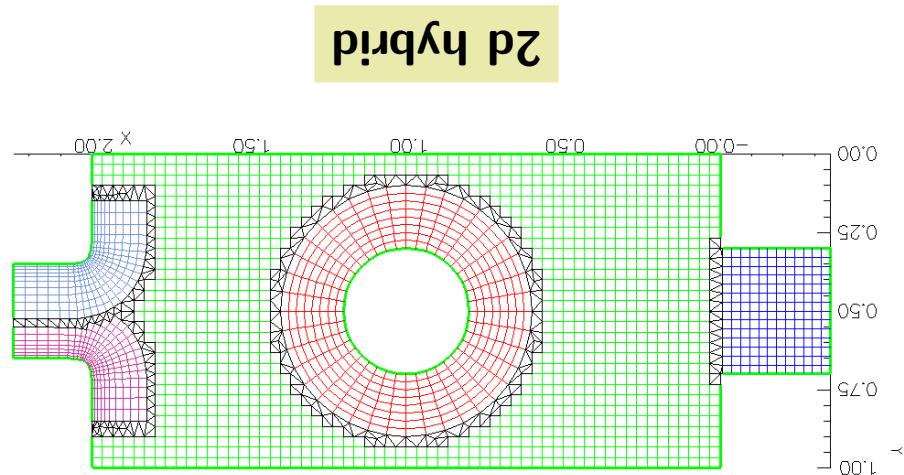
Goal: build high-quality unstructured grids that are contain large structured regions.

Key features:

- ◊ two and three dimensional hybrid grids,
- ◊ advancing front algorithm,
- ◊ mesh quality improvements



3D hybrid grid



2d hybrid

- A Short History of Composite/ Chimera/ Overset/ Overlapping Grids
- Volkov, [1966] developed a Composite Mesh method for Laplace's equation on regions with piece-wise smooth boundaries separated by corners. Polar grids are fitted around each corner to handle potential singularities.
 - Starius, [1977] (student of H.-O. Kreiss) considered Composite Mesh methods for elliptic and hyperbolic problems – introduces a hyperbolic grid generator.
 - Steger, circa [1980] independently conceives the idea of the overlapping grid, subsequently named the Chimera approach after the mythical Chimera beast having a human face, a lion's mane and legs, a goat's body, and dragon's tail. NASA groups develop grid generator PEGSUS, hyperbolic grid generation and flow solver Overflow (Steger, Benek, Suh, Buning, Chan, Meakin, et. al.)
 - B. Kreiss [1980] develops overlapping grid generator which subsequently leads to the CMGRD grid generator [1983] (Cheshire, Henshaw) later leading to the Overture set of tools [1994].

Composite/ Chimera/ Overset/ Overlapping Grids

Reference Henshaw (JCP,1994); Henshaw, Kreiss, and Reyna (Comput. Fluids,1994); Karniadakis, Israeli and Orszag (JCP,1991); Petersson (JCP, 2001);

$$\frac{\partial n}{\partial p} = \mathbf{u} \cdot (-\nabla \Delta \times \Delta \times \mathbf{n})$$

On no-slip walls we use the *numerical pressure boundary condition*:

$$\begin{aligned} \mathbf{x} \in \partial \Omega & \quad \left\{ \begin{array}{lcl} 0 & = & \Delta \cdot \mathbf{n} \\ 0 & = & B(\mathbf{n}, p) \end{array} \right. \quad \text{("pressure BC" } \longleftarrow \text{)} \\ \mathbf{v} \ni \mathbf{x} & \quad \left\{ \begin{array}{lcl} 0 & = & \mathbf{n} \cdot \Delta(\mathbf{x}) - \mathbf{n} \Delta : \mathbf{n} \Delta + d \nabla \\ \mathbf{n} \nabla \mathbf{v} & = & d \Delta + \mathbf{n} (\Delta \cdot \mathbf{n}) + \mathbf{n} \end{array} \right. \end{aligned}$$

Velocity-pressure formulation:

- ◊ supports moving grids (e.g. motion of rigid bodies).
- ◊ accurate and stable boundary conditions.
- ◊ second and fourth-order accurate in space and time.
- ◊ implicit and explicit time-stepping (method of lines).

pressure equation.

- ◊ Split-step, pressure-Poisson formulation: advance the velocity first, then solve the

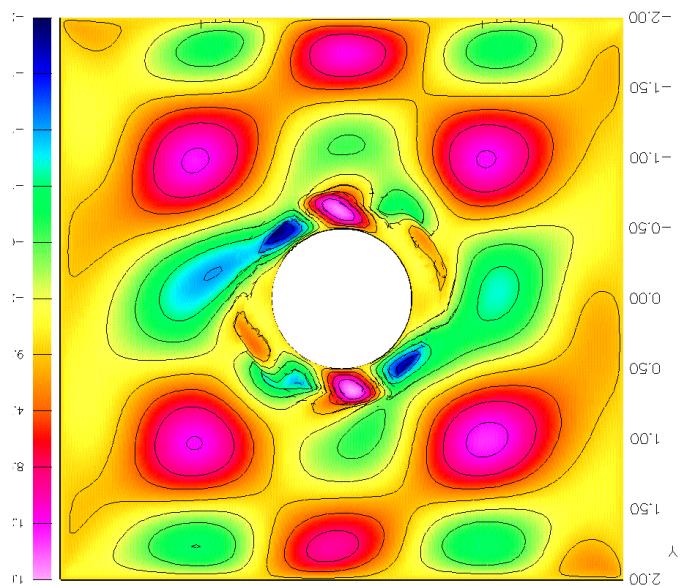
OverBlow'n's Incompressible Navier Stokes Solver

$$t = 1.$$

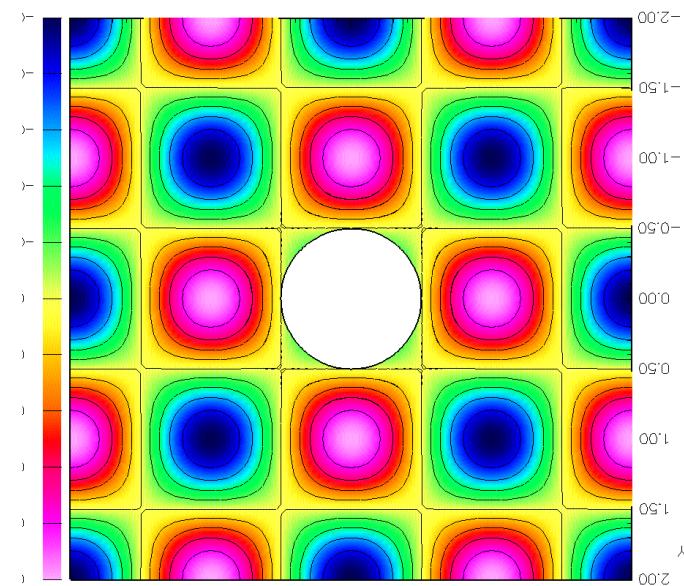
Maximum errors in the computed solution and convergence rate, $e \propto h^\alpha$, $\nu = 0.1$,

grid	N	p	u	v	$\nabla \cdot u$	rate α
cicb.order4	61	$1.1e-4$	$3.5e-5$	$3.4e-5$	$1.9e-4$	3.8
cic.order4	121	$8.9e-6$	$1.5e-6$	$2.0e-6$	$1.7e-5$	4.3
cic2.order4	241	$6.0e-7$	$1.0e-7$	$1.2e-7$	$1.0e-6$	4.1
						3.8

Error in u



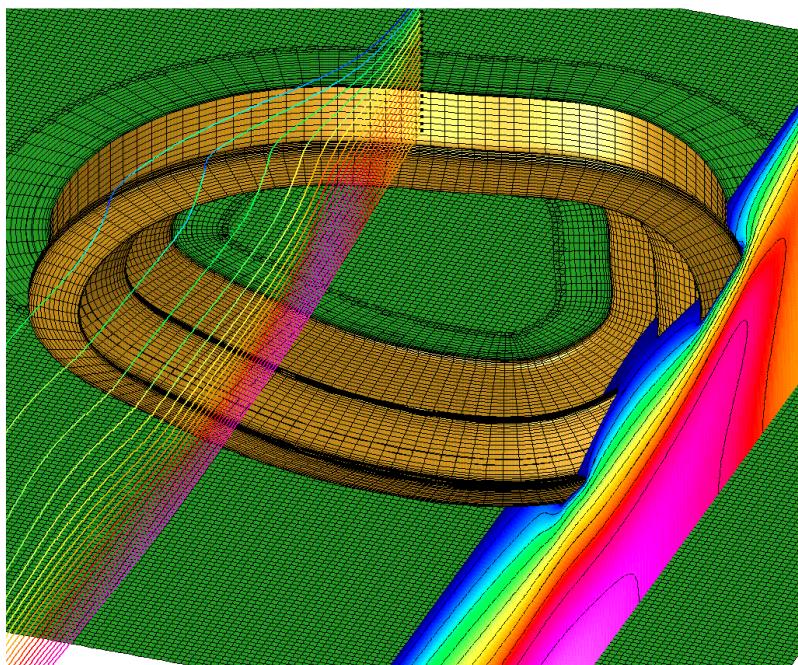
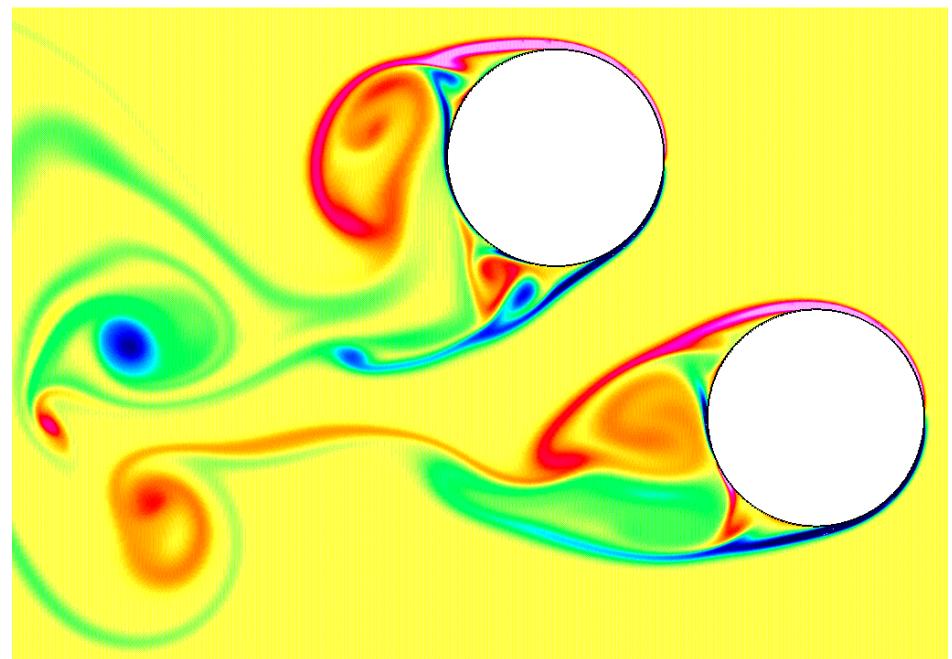
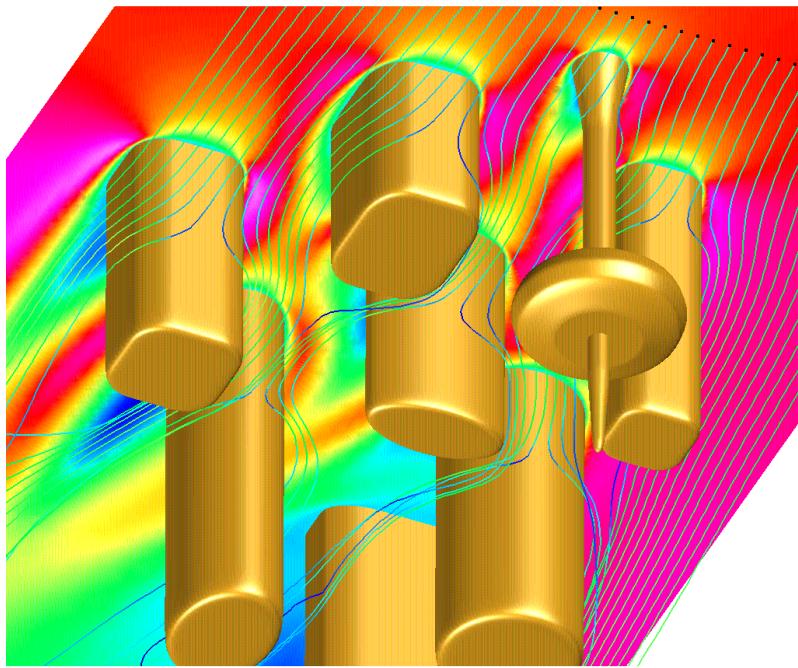
Computed solution u



analytic solutions

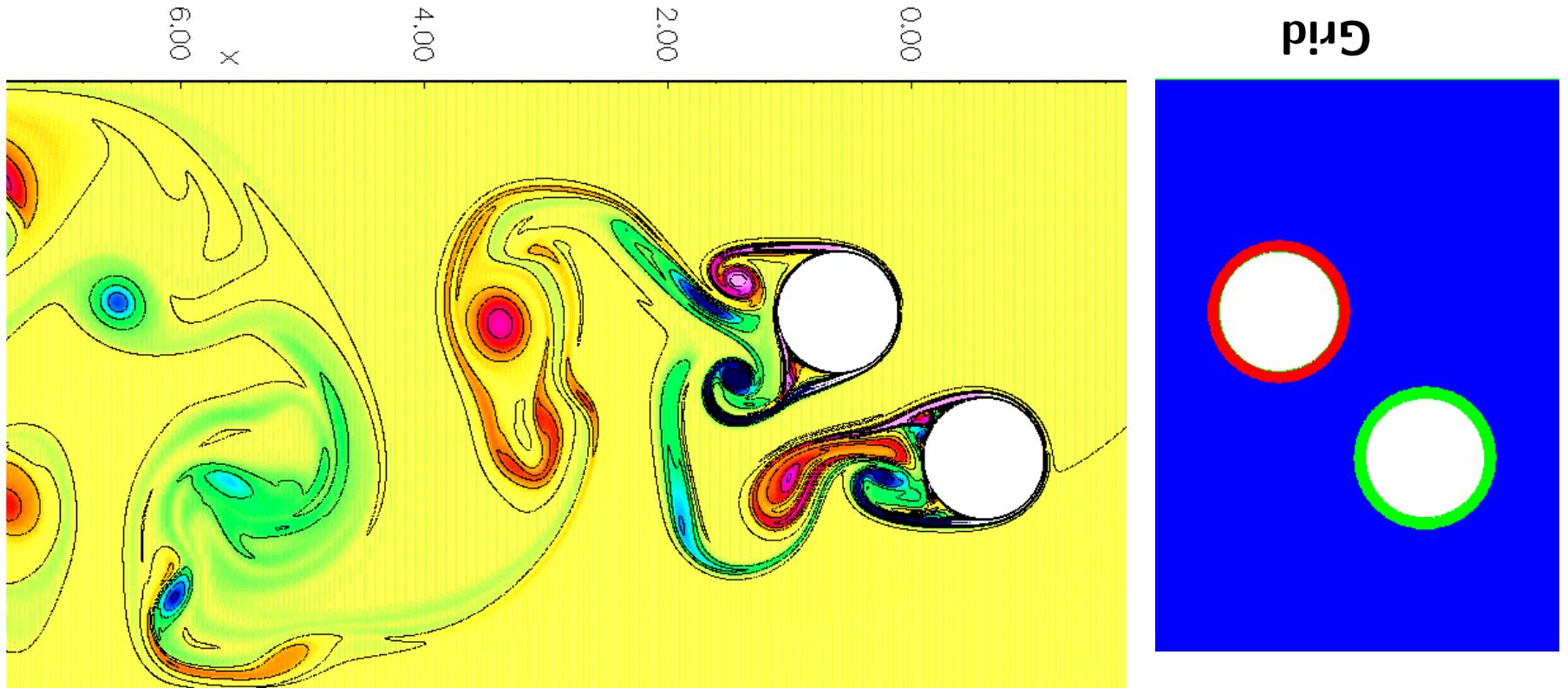
Incompressible Navier-Stokes: testing with the method of

Incompressible flow computations with OverBlown.



- ◊ multigrid solution of the pressure and implicit time-stepping equations,
- ◊ requires 460 MB of memory,
- ◊ 1.3 million grid points,
- ◊ cpu = 10 s/step,
- ◊ 2.2 GHz Xeon, 2 GB of memory

Flow past two cylinders, vorticity. $Re_D=2000$.

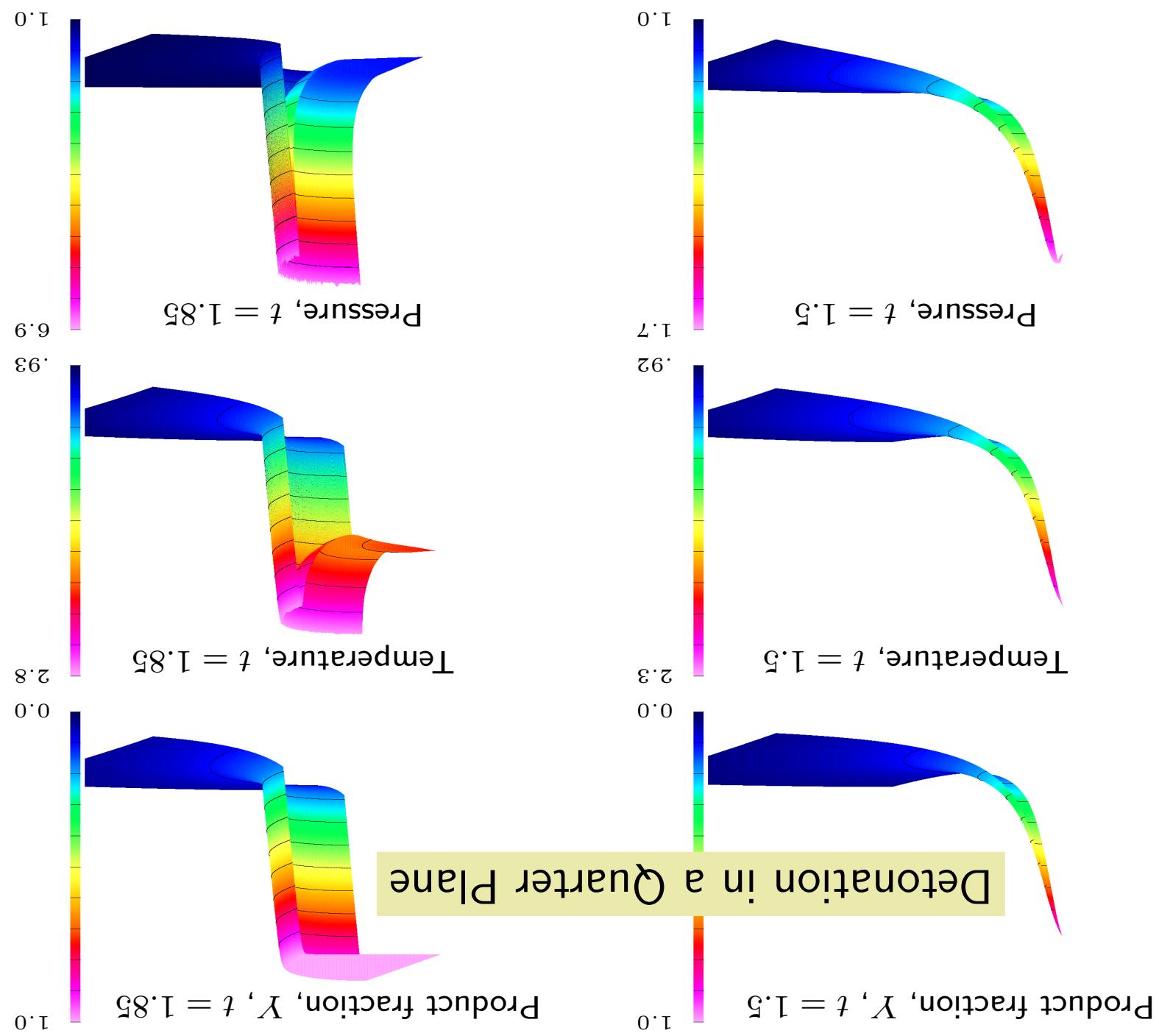


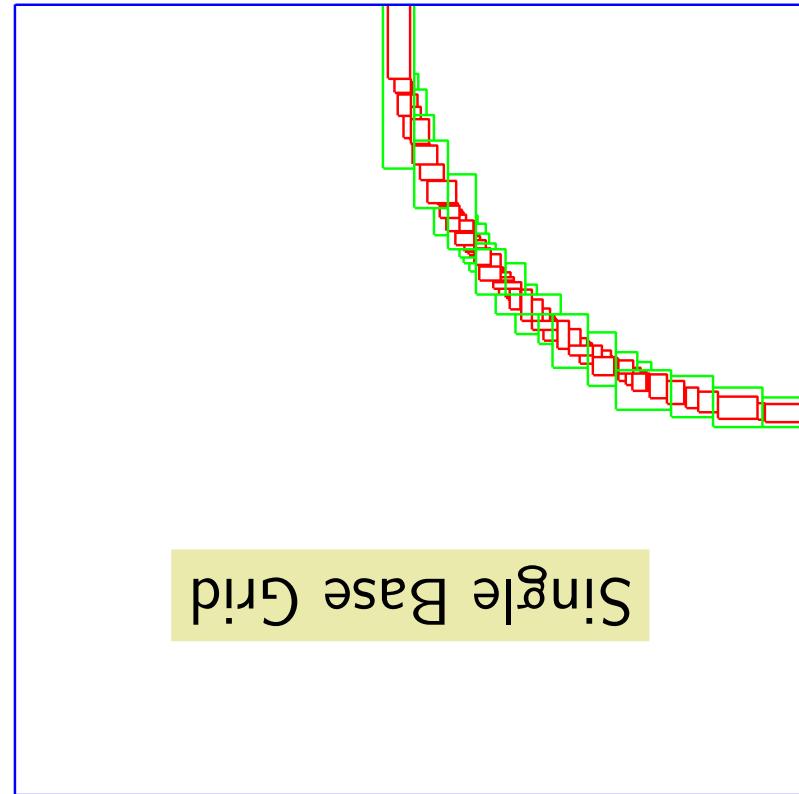
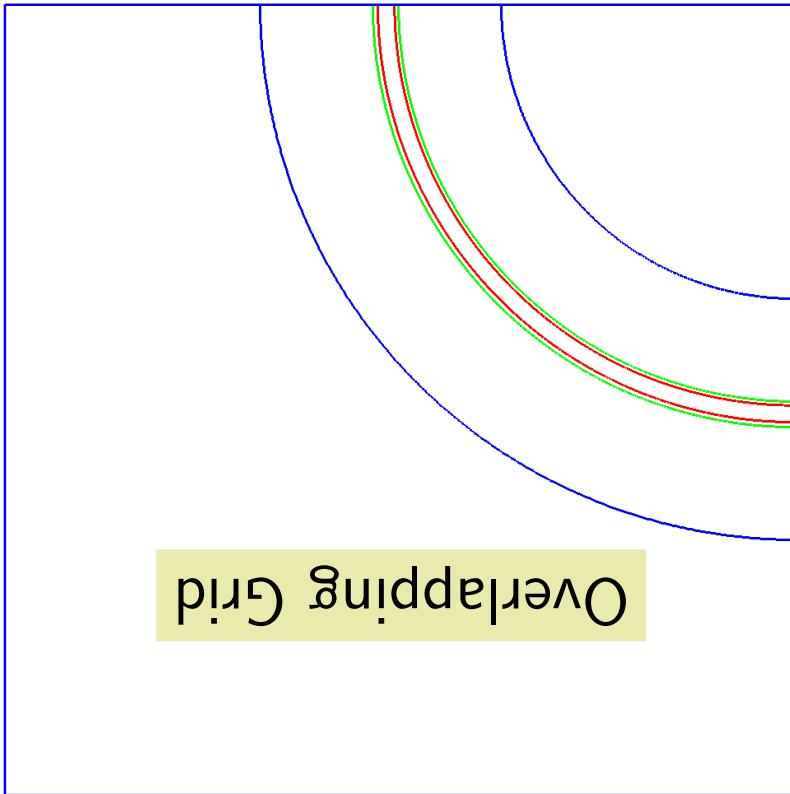
Reference: Henshaw and Schwendeman An Adaptive Numerical Scheme for High-Speed Reactive Flow on Overlapping Grids, JCP vol. 191, 2003.

$$\begin{bmatrix} \rho F \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \rho u \chi \\ (d + E)u \\ \rho u^2 + d \\ \rho u v \\ \rho u \end{bmatrix} + \begin{bmatrix} \rho u \chi \\ (d + E)u \\ \rho u v \\ \rho u^2 + d \\ \rho u \end{bmatrix} + \begin{bmatrix} \rho \chi \\ E \\ \rho u \\ \rho u \\ \rho \end{bmatrix}$$

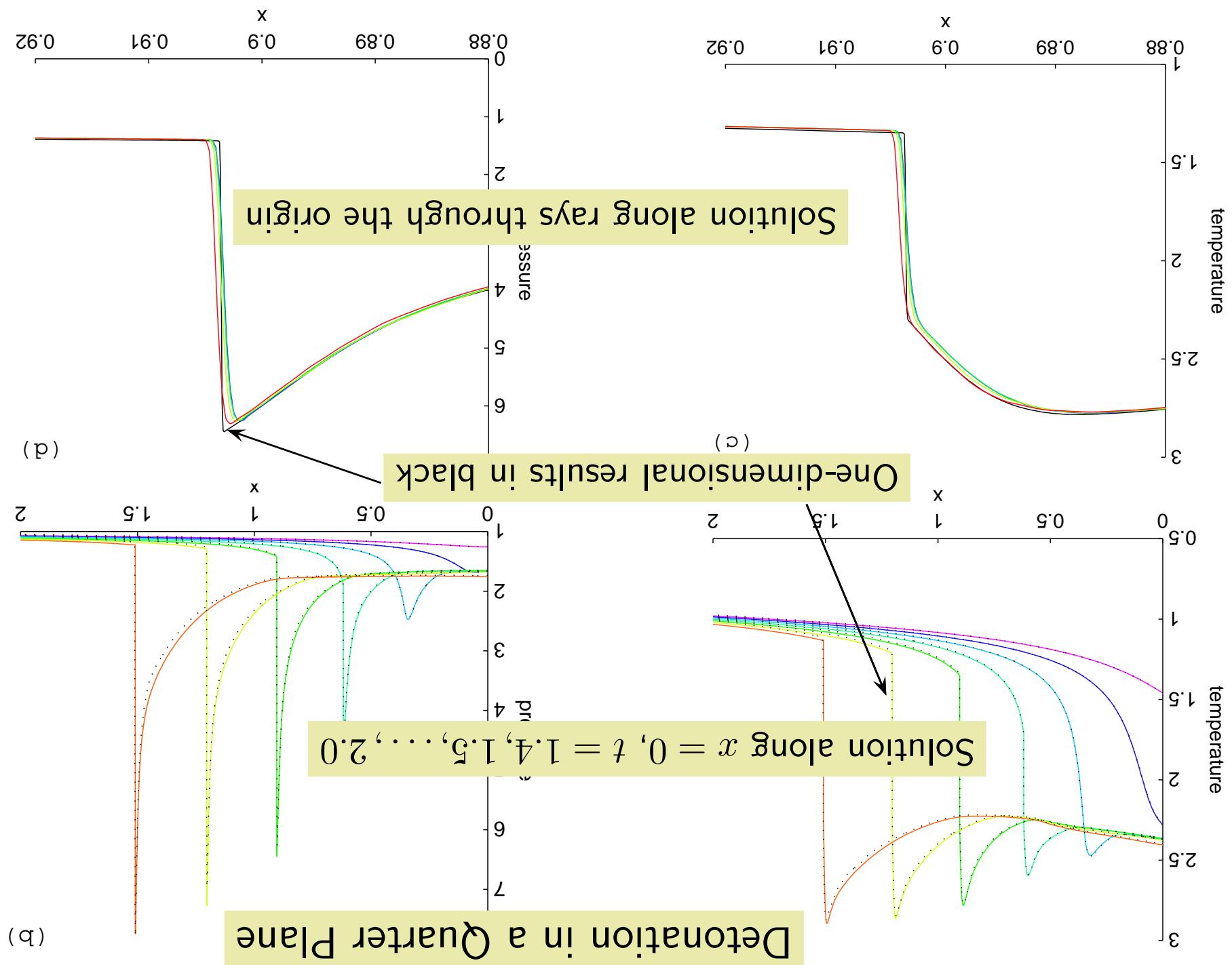
- In progress, a multiphase (Baeer-Nunziato like) formulation.
- In progress, a multi-fluid formulation (Jeffrey Banks, RPI).
- Equations of state, ideal Gas, JWL, Mie-Grueneisen.
- Reaction mechanisms: one-step, chain branching, ignition-and-growth;
- moving grids and AMR (in 2D)
- High-order Godunov method (Don Schwendeman, RPI)

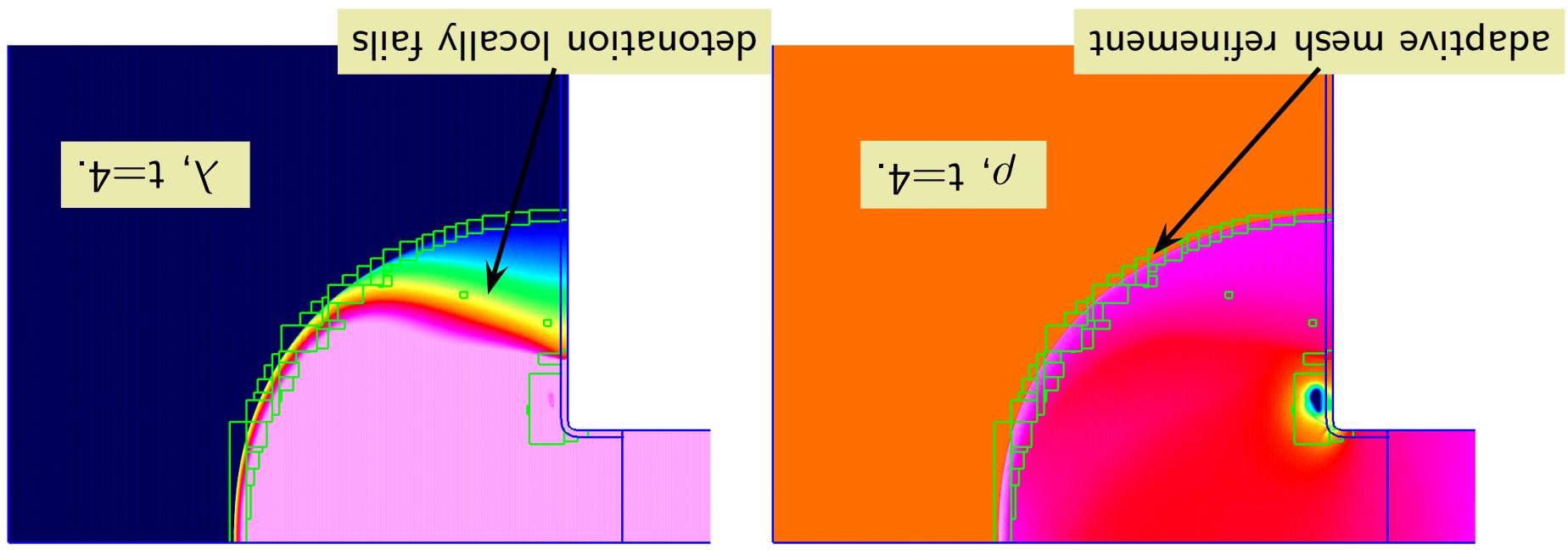
OverBlow's Reactive Euler Equations Solver



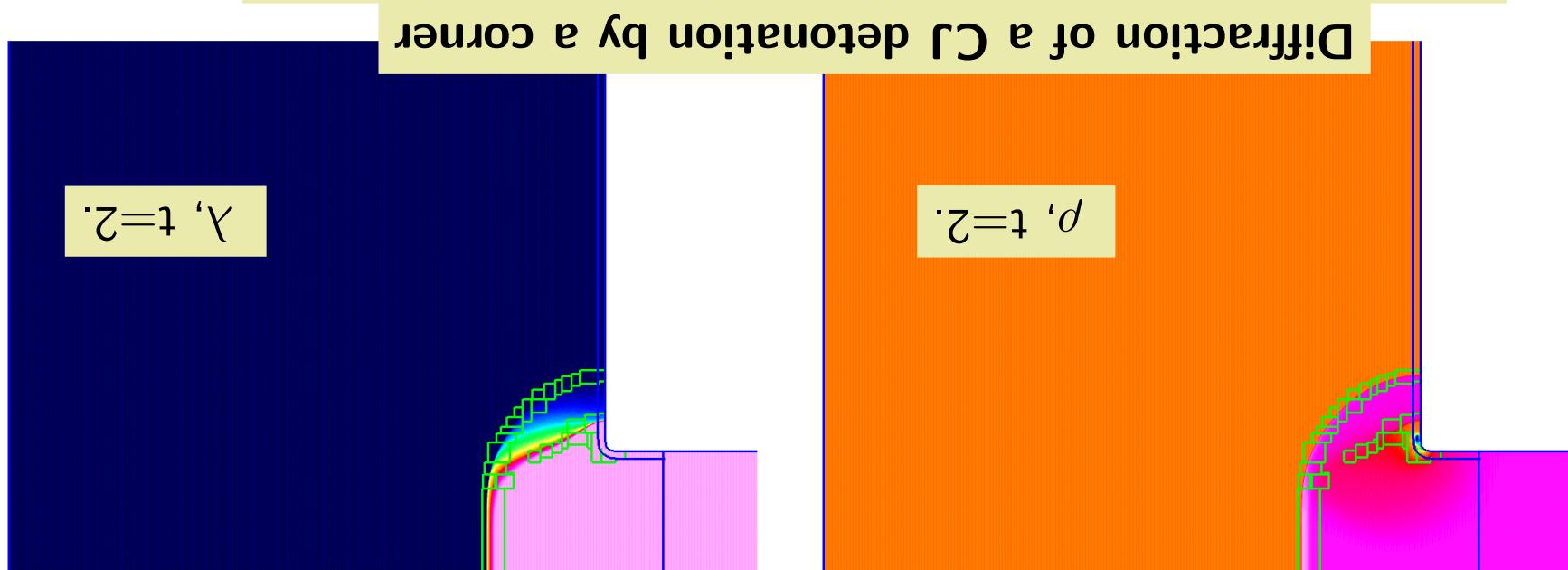


Detonation in a Quarter Plane





LX-17, ignition-and-growth model, JWL equation of state.



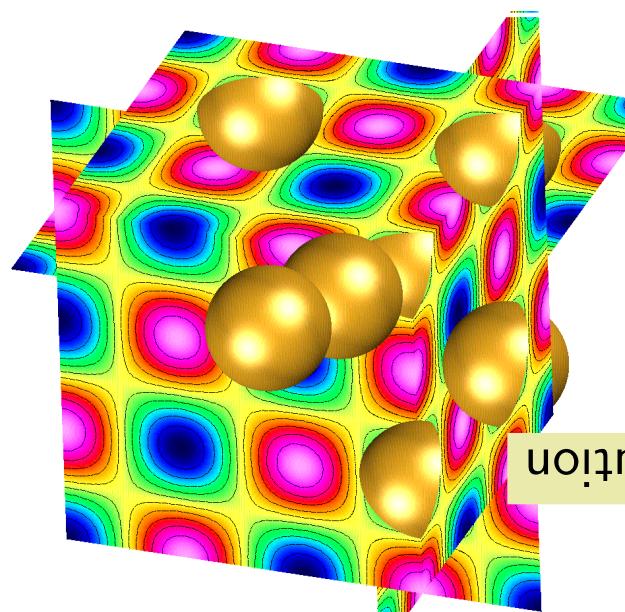
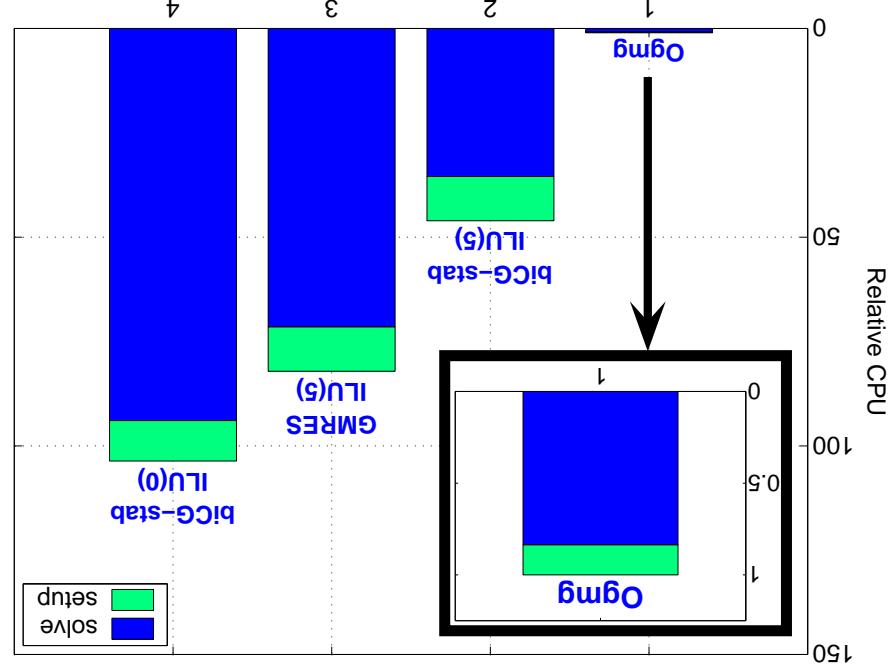
A new multigrid algorithm for overlapping grids has been developed, implemented in the O_{gmg} solver. It can be used to solve scalar elliptic boundary value problems. Some of its key features are coarse grid generation - robust generation of „any“ number of multigrid levels, adaptive smoothers, variable sub-smoothers per component grid, interpolation-boundary smoothing, over-relaxed Red-Black smoothers, Galerkin coarse grid operators, numerical boundary conditions for Dirichlet and Neumann/mixed problems and speed and memory optimizations for Cartesian grids and pre-defined quadratures.

Reference Henshaw [SIAM J. Sci. Comput., 2005] (to appear).

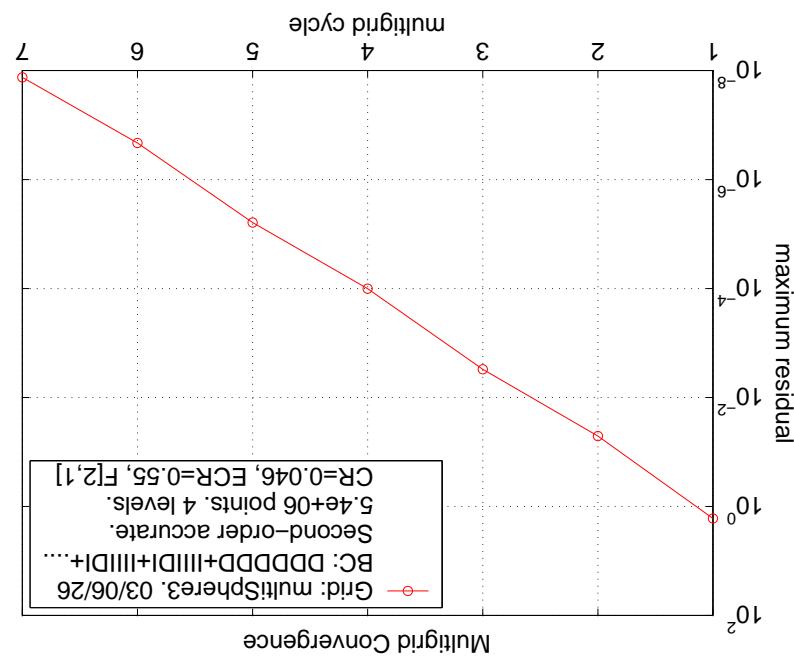
- ◊ speed and memory optimizations for Cartesian grids and pre-defined quadratures
- ◊ numerical boundary conditions for Dirichlet and Neumann/mixed problems
- ◊ Galerkin coarse grid operators
- ◊ over-relaxed Red-Black smoothers
- ◊ interpolation-boundary smoothing
- ◊ variable sub-smoothers per component grid
- ◊ adaptive smoothers
- ◊ coarse grid generation - robust generation of „any“ number of multigrid levels.

A Multigrid Algorithm for Overlapping Grids

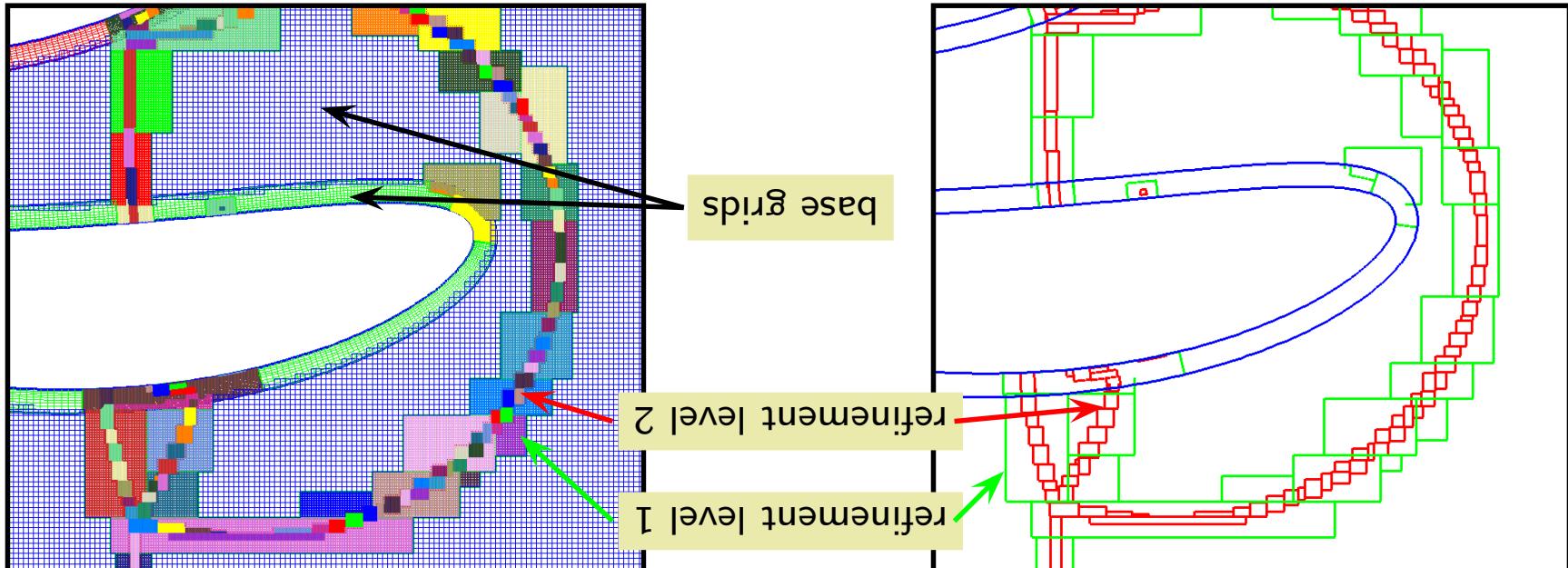
Multigrid solution to Poisson's equation, 5.4 million grid points



Mesg independent convergence rates

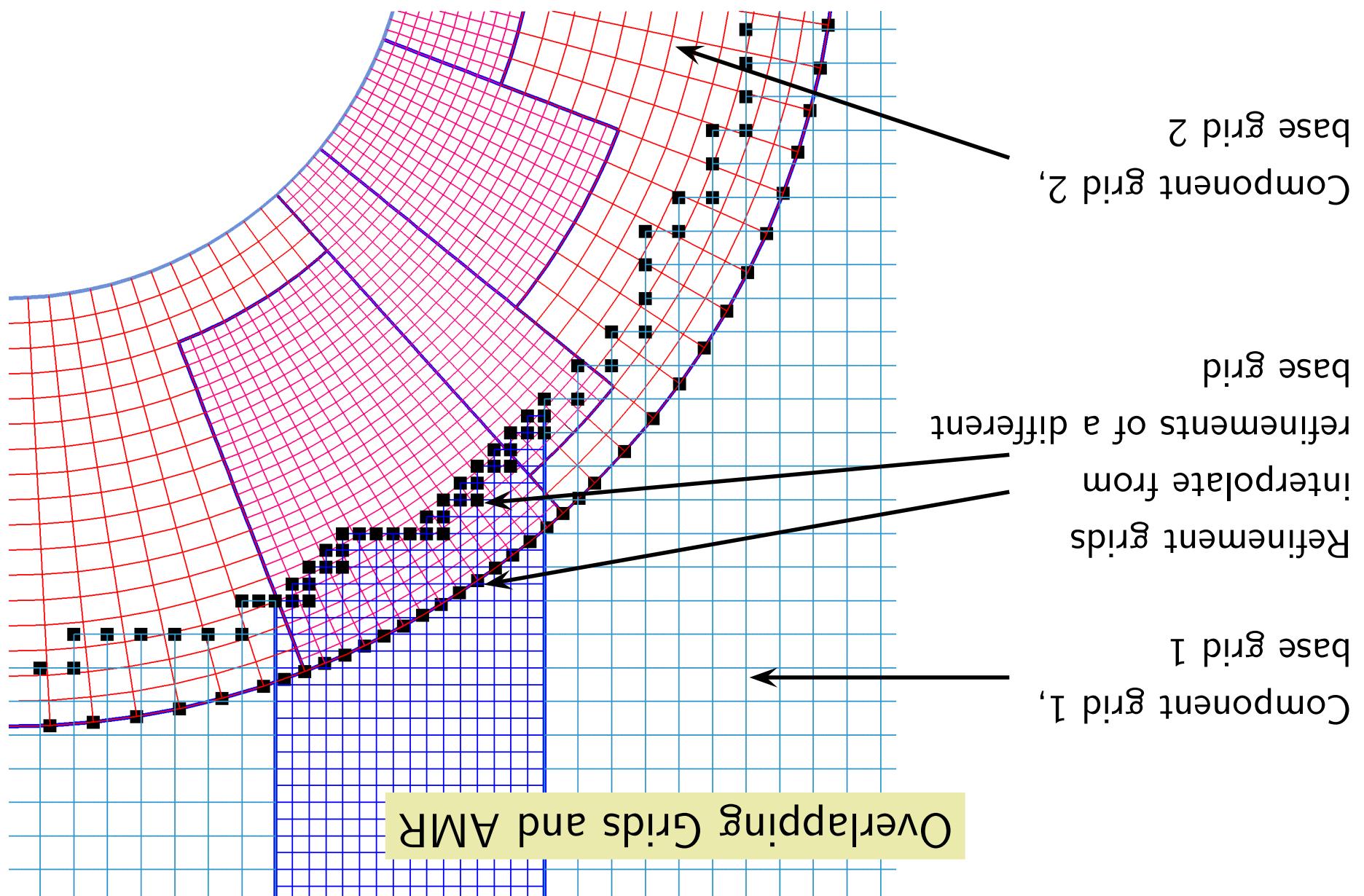


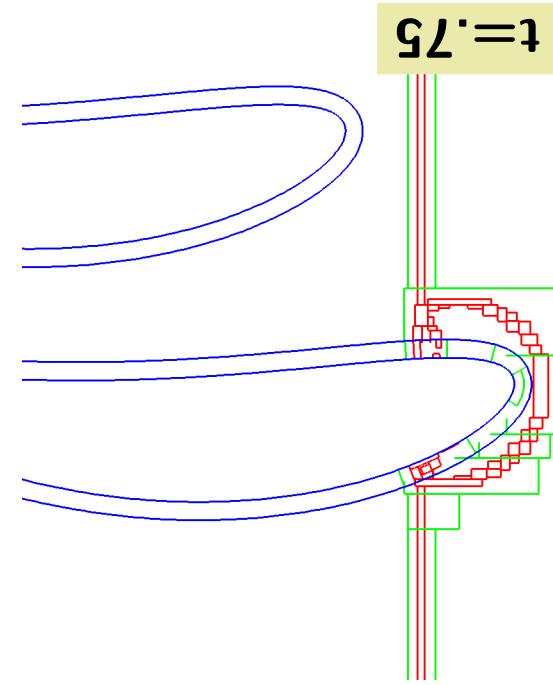
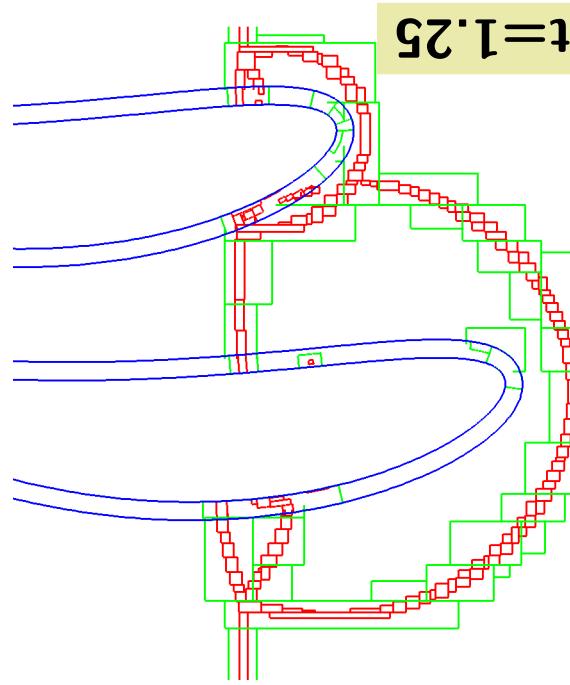
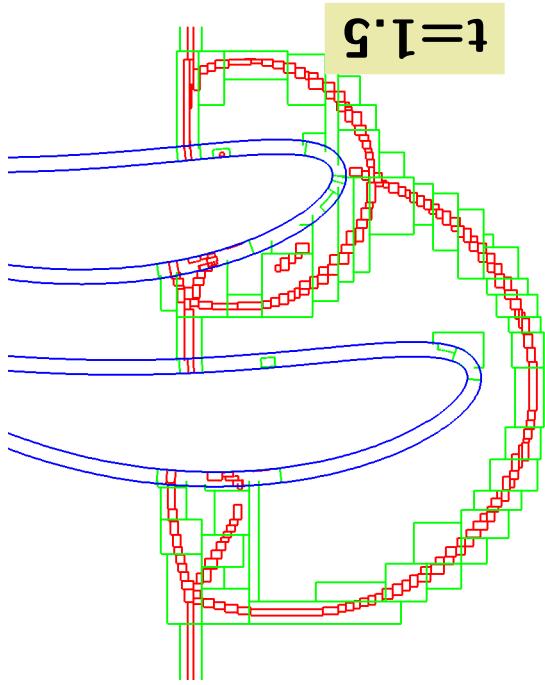
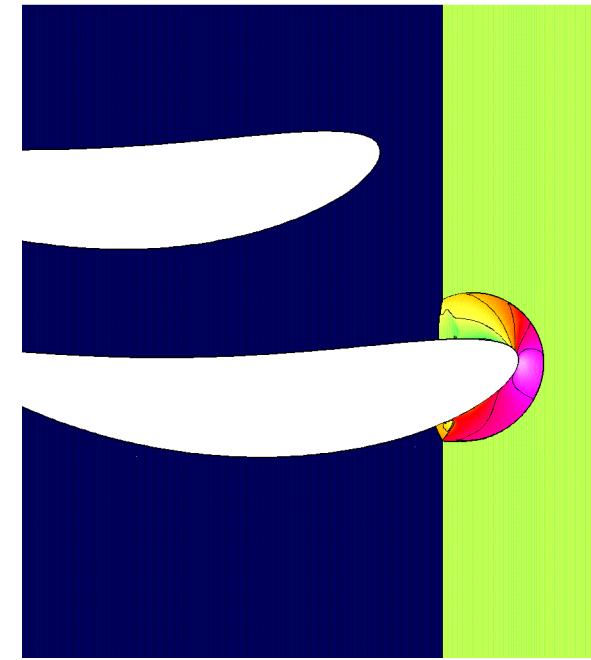
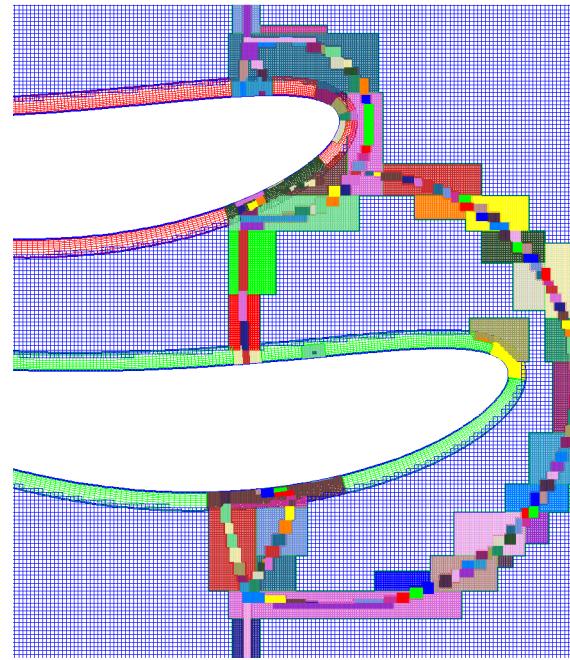
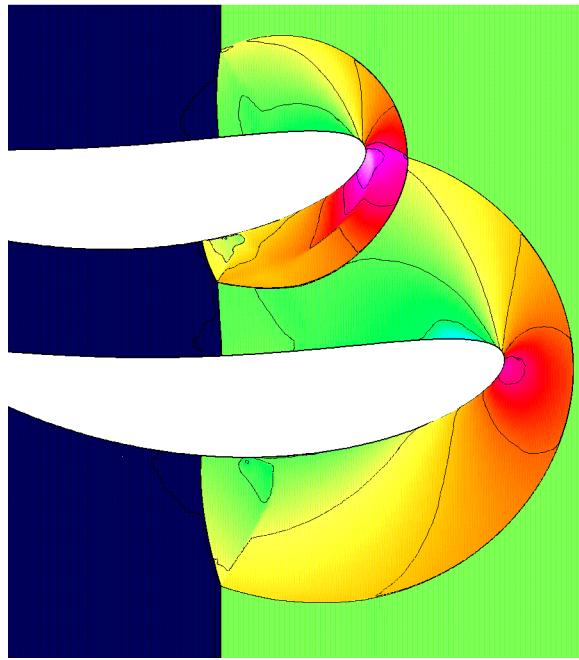
In comparison to Krylov solvers multigrid is an order of magnitude faster and uses an order of magnitude less storage



- ◊ Refinement patches are generated in the parameter space of each component grid (base grid).
- ◊ Refinement patches are organized in a hierarchy of refinement levels.
- ◊ Error estimators determine where refinement is needed.
- ◊ AMR grid generation (Bergger-Rigoutsos algorithm) builds refinement patches based on the error estimate.
- ◊ refinement grids may interpolate from refinement grids of different base grids.
- ◊ The key issue is efficiency.

Block Structured Adaptive Mesh Refinement and Overlapping Grids

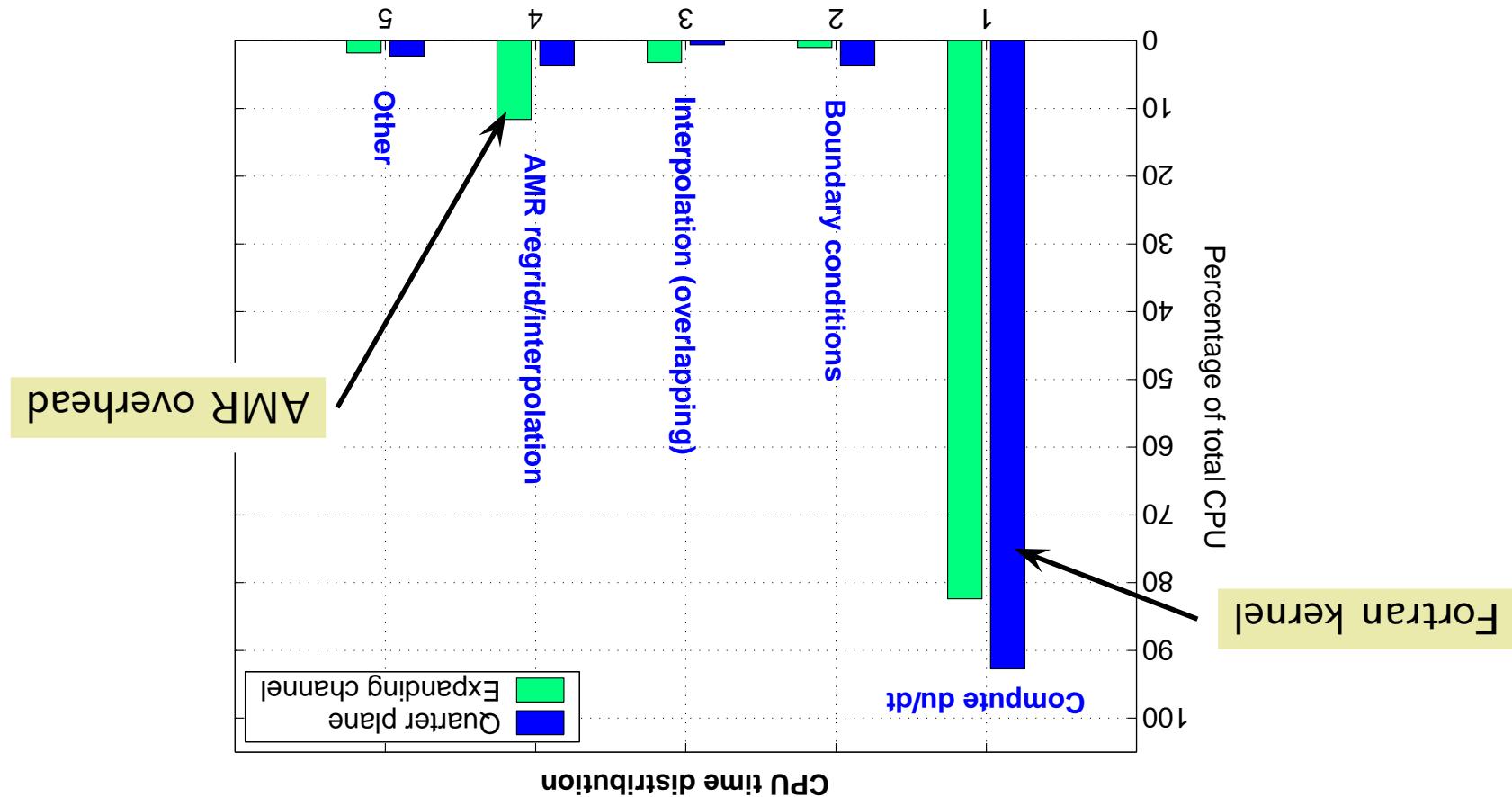




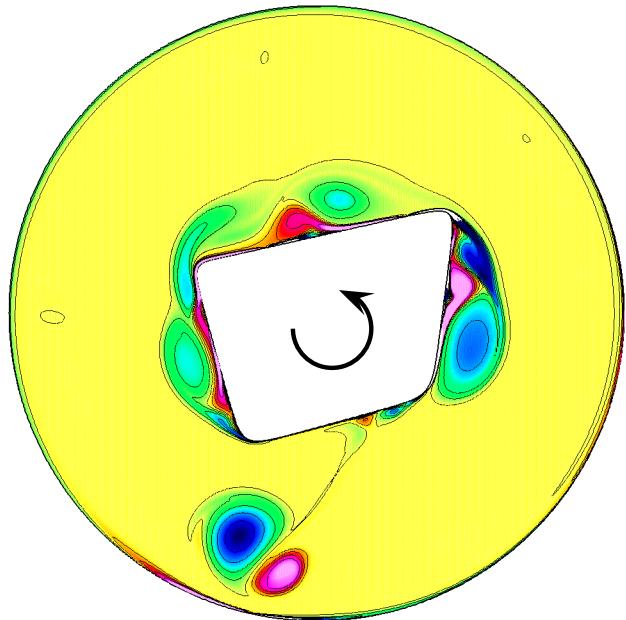
Adaptive overlapping grids

Overlapping grid AMR performance on two detonation problems.

time steps	21030	12418	14.94	13.96	274, 588	(2, 57, 353)	(2.0e5, 9.2e5, 1.9e6)	(1.2e5, 6.4e5, 1.3e6)	points (min,ave,max)
grids (min,ave,max)									
seconds per step									



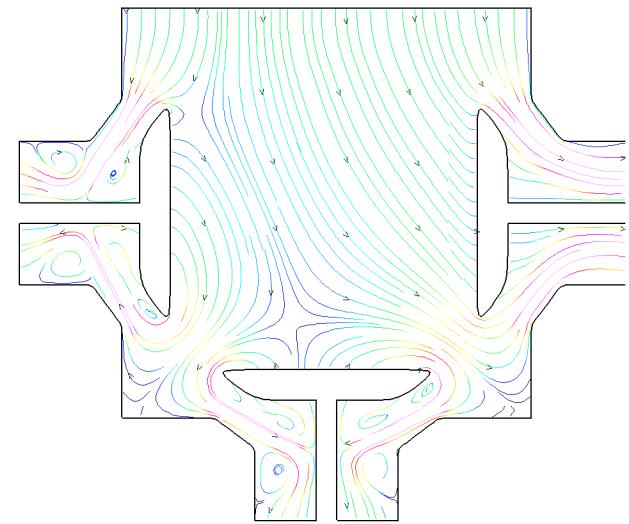
Rotating body (INS)



Falling cylinders (INS)



Moving valves (INS)



- ◊ Boundary fitted component grids are used to discretize each moving body.
- ◊ Grids move at each time step according to some governing equations.
- ◊ Overlapping connectivity information is updated by Qgen (interpolation points, discretization points, unused points).
- ◊ Solution values at **exposed points** are interpolated at previous time levels.
- ◊ Detection and treatment of collisions – elastic/in-elastic collisions
- ◊ Issue: Bodies that get very close – how should the grids interpolate

Moving Overlapping Grids

```

    }  

    end  

    applyBoundaryConditions( $\mathcal{G}$ ,  $u_n^i$ ,  $t$ );  

    interpolate( $\mathcal{G}$ ,  $u_n^i$ );  

     $t := t + \Delta t$ ;  $\mathcal{G} := \mathcal{G}_{n+1}$ ;  $n := n + 1$ ;  

     $u_{n+1}^i := \text{advanceTimeStep}(\mathcal{G}, \mathcal{G}_{n+1}, u_n^i, \Delta t)$ ;  

     $\mathcal{G}_{n+1} := \text{moveGrids}(\mathcal{G}, u_n^i)$ ;  

end  

 $g := g_*$ ;  

 $u_*^i := \text{interpolateToNewGrid}(u_n^i, \mathcal{G}, g_*)$ ;  

 $g_* := \text{regrid}(\mathcal{G}, e_i)$ ;  

 $e_i := \text{estimateError}(\mathcal{G}, u_n^i)$ ;  

if ( $n \bmod n_{\text{regrid}} = 0$ ) then  

  while  $t < t_{\text{final}}$   

     $u_n^i := \text{applyInitialCondition}(\mathcal{G})$ ;  

     $t := 0$ ;  $n := 0$ ;  

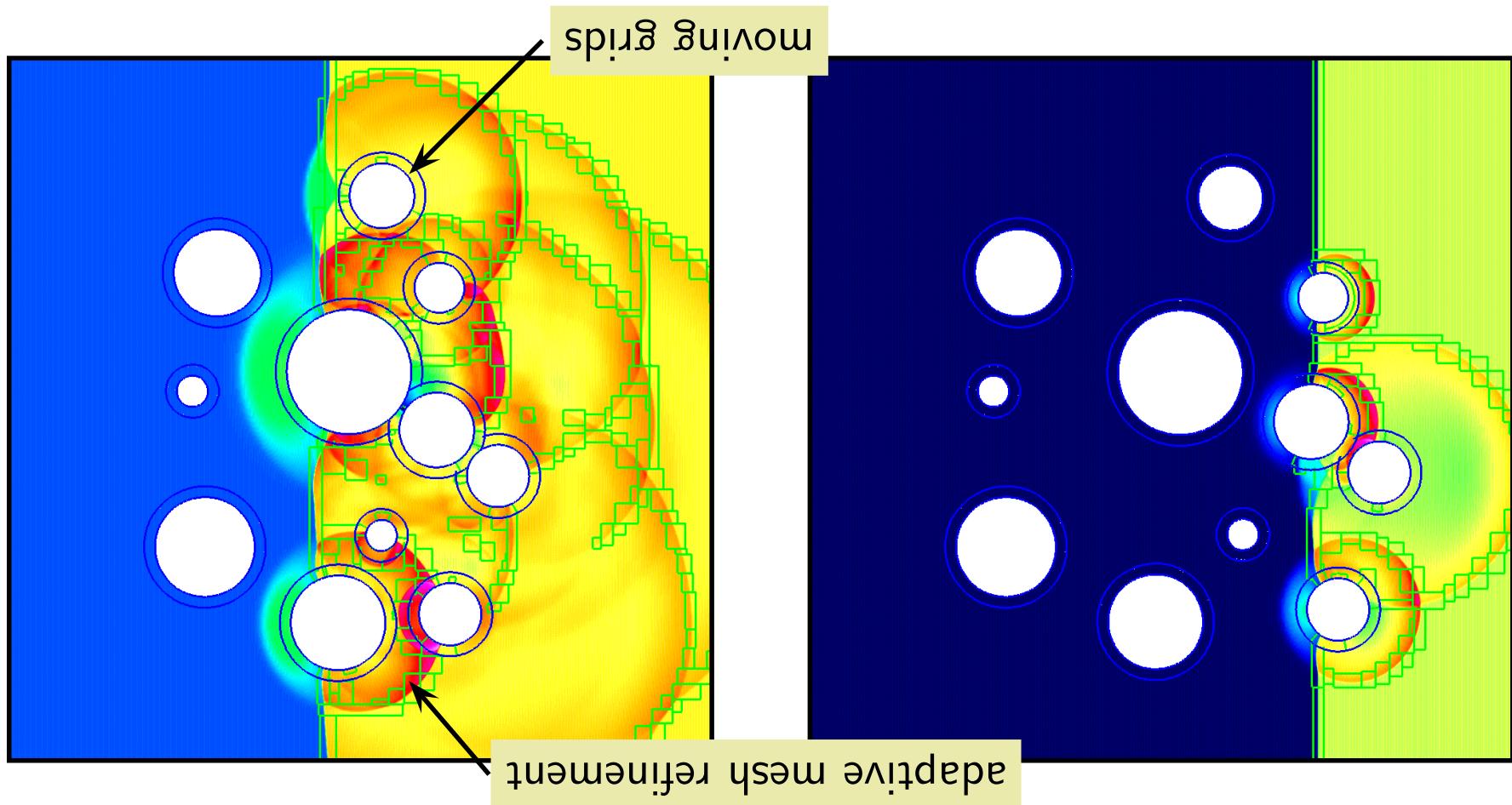
  end  

PDESolve( $\mathcal{G}, t_{\text{final}})$ 
}

```

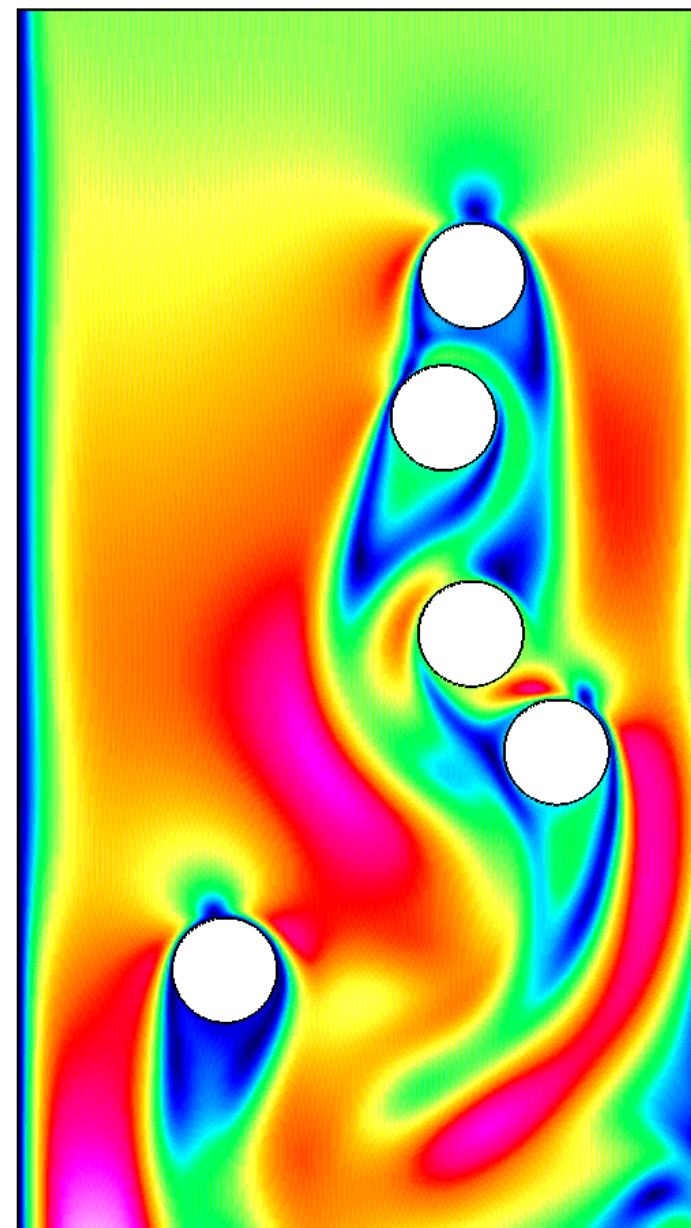
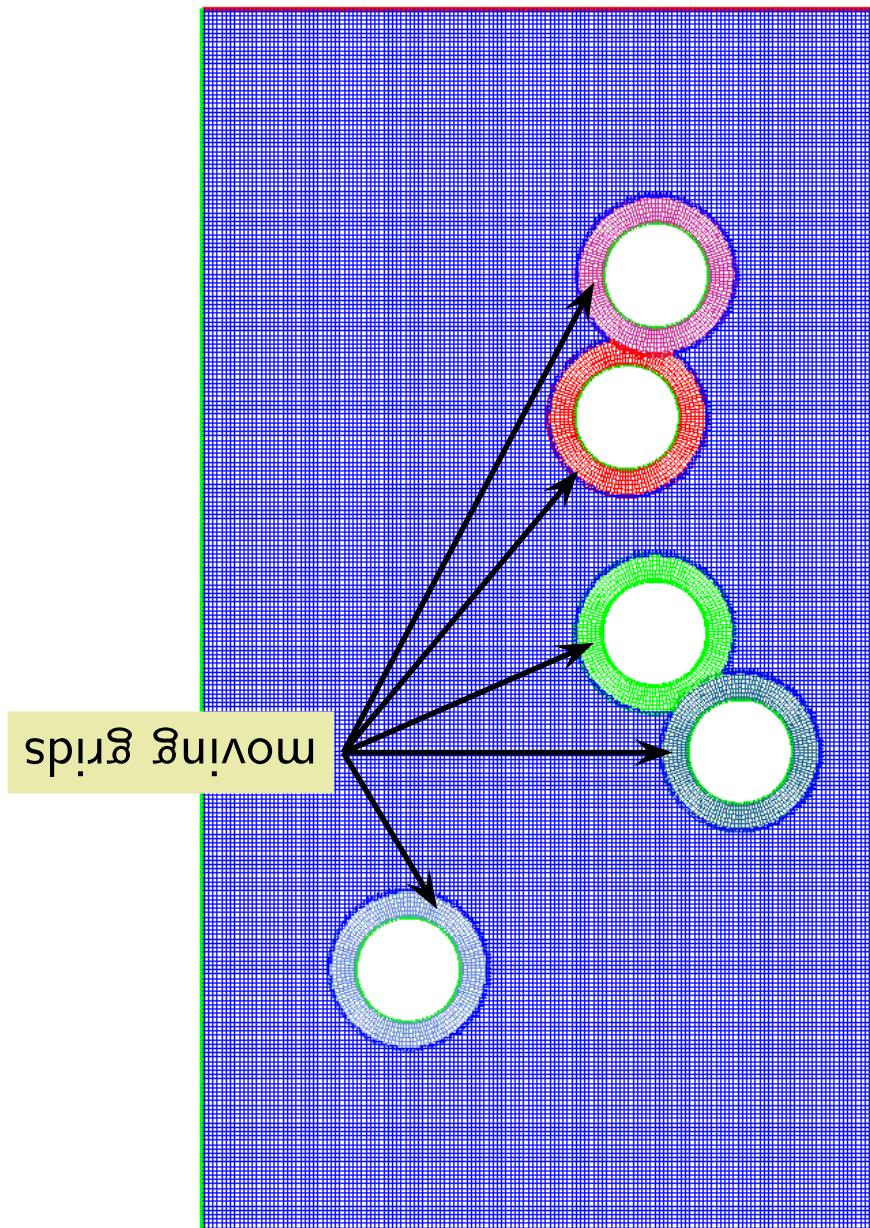
The Basic AMR Moving-Grid Time Stepping Algorithm

A shock hitting a collection of cylinders (density).

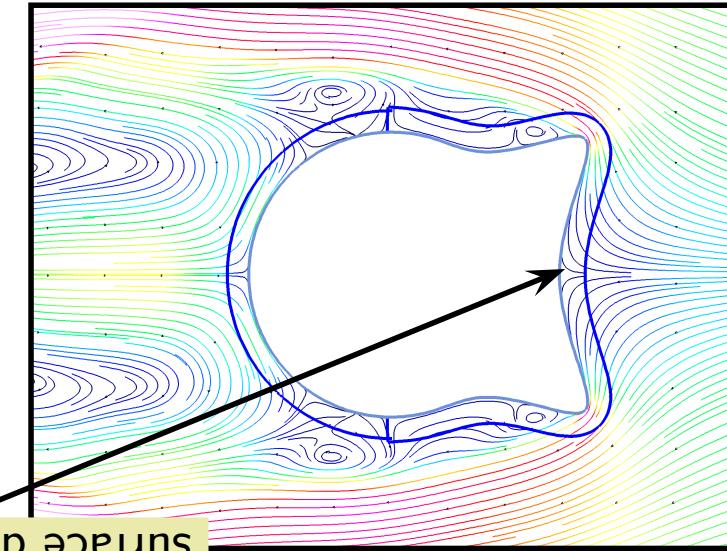
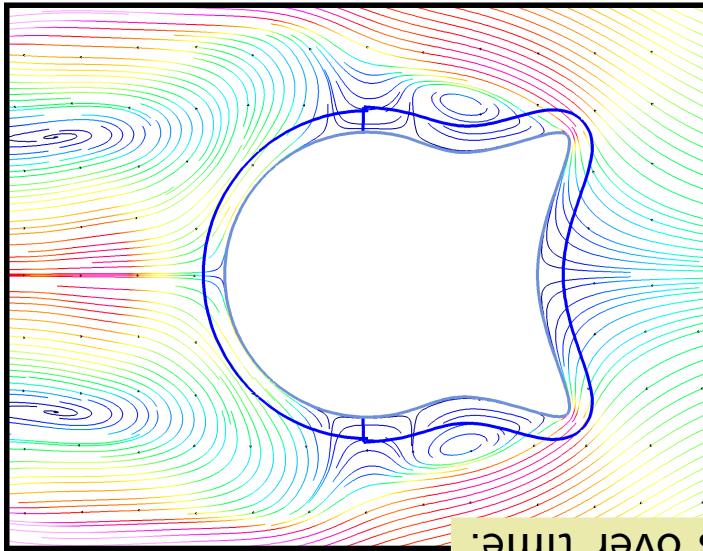


Moving geometry and AMR

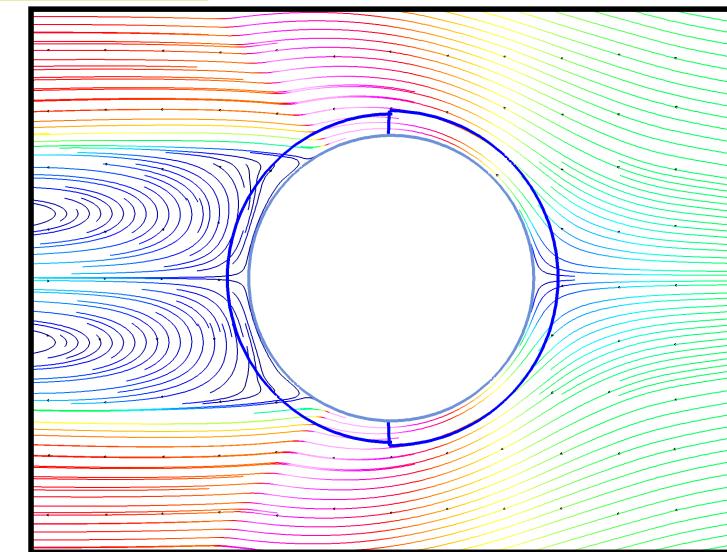
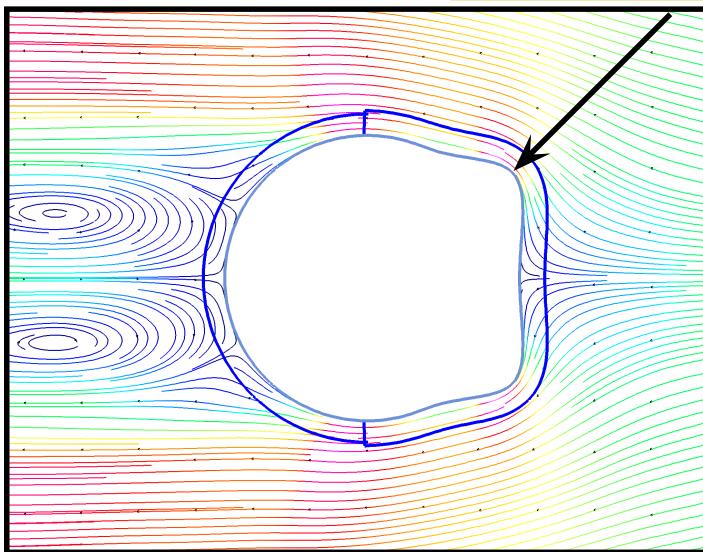
Falling cylinders in an incompressible flow



Streamlines of a compressible flow around a deforming boundary.



surface deforms over time.



Modeling Deforming Geometry with Overlapping Grids

The model for distributed parallel computing in Overture

- ◊ Grids can be distributed across one or more processors.
- ◊ Distributed parallel arrays using P++ (K. Brislawm, B. Miller, D. Quinlan)
- ◊ P++ uses Multiblock PARTI (A. Sussman, G. Agrawal, J. Saltz) for block structured communication with MPI (ghost boundary updates, copies between different distributed arrays)
- ◊ A special parallel overlapping grid interpolation routine is used for overlapping grid interpolation.

```

P++ : parallel multi-dimensional arrays

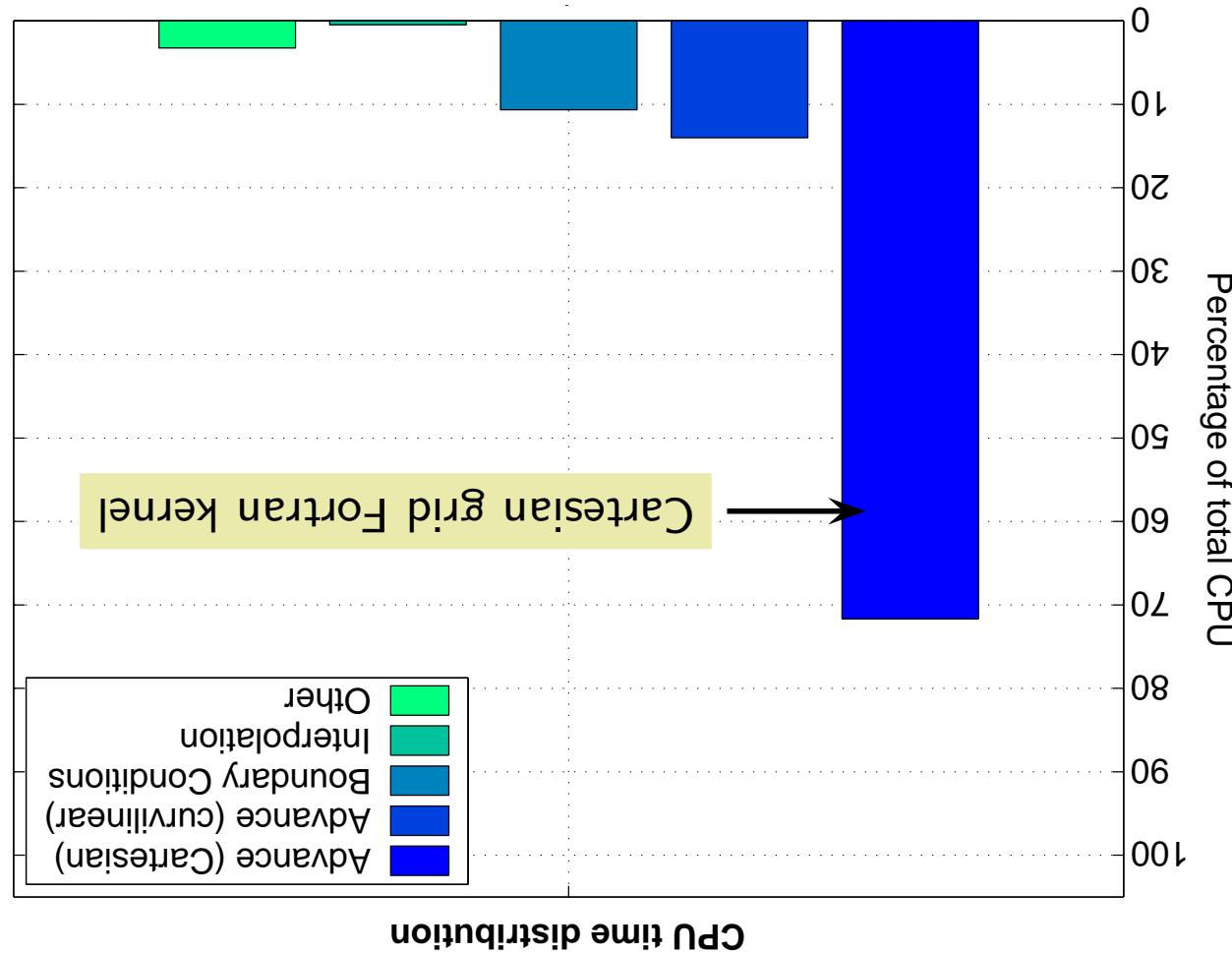
Partitioning_Type partition; // object that defines the parallel distribution
partition.SpecifyInternalGhostBoundaryWidths(1,1);
Paritioning_Type partition; // object that defines the parallel distribution
partition.SpecifyInternalGhostBoundaryWidths(1,1);
parallelDistributedArray u(100,100,partition); // build a distributed array
Range I(1,98), J(1,98);
// Parallel array operation with automatic communication:
u(I,J)=.25*( u(I+1,J) + u(I-1,J) + u(I,J+1) + u(I,J-1) ) + sin(u(I,J))/3.;

// Access local serial arrays and call a Fortran routine:
realISerialArray & uLocal = u.getLocalArray(); // access the local array
myFortranRoutine(*uLocal.getDataPointer()....);
u.updateGhostBoundaries(); // update ghost boundaries on distributed arrays

```

Two-dimensions, 3.8 million grid-points.

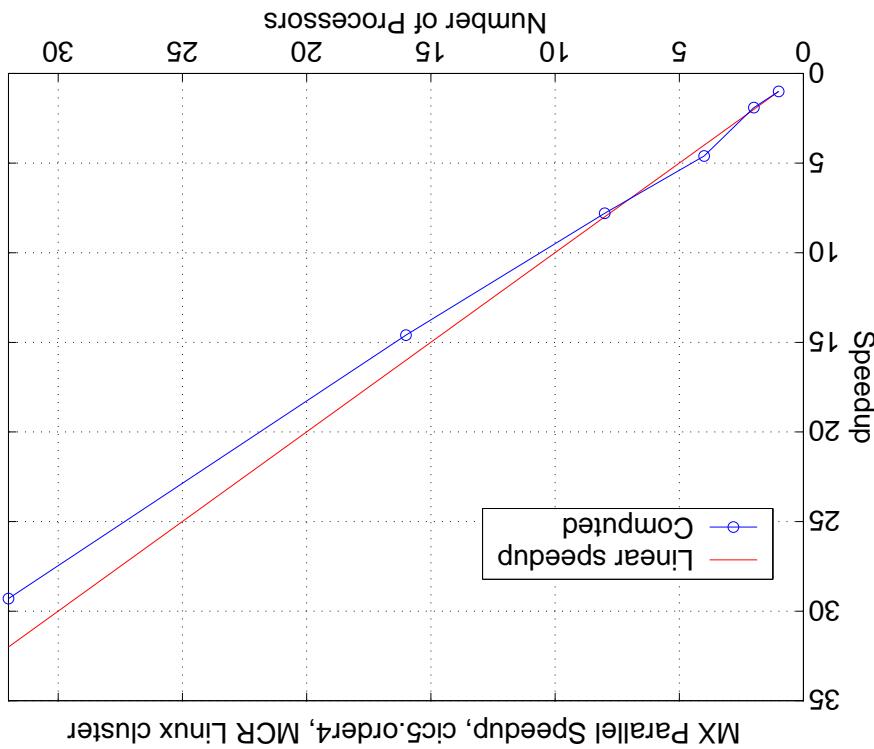
Performance of the Maxwell solver (serial).



of Cartesian grid codes.

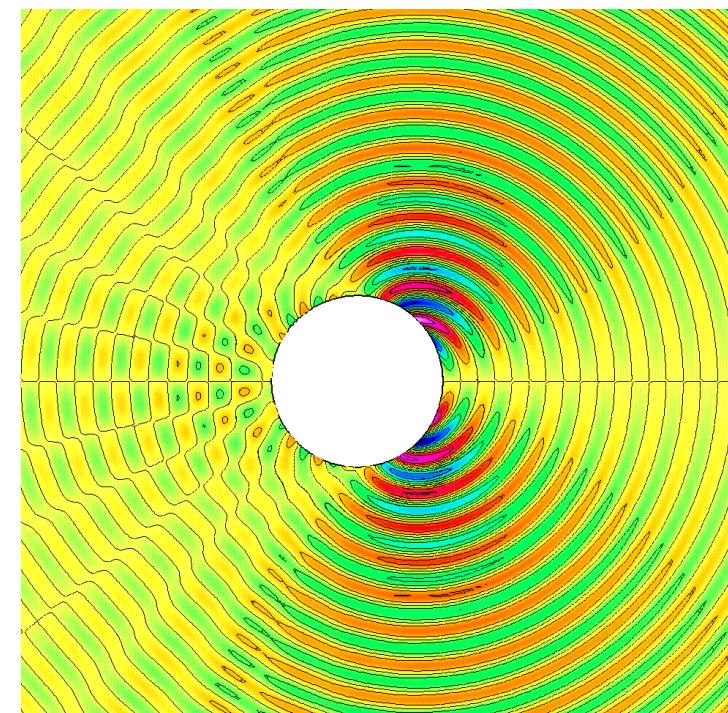
Performance of overlapping grid codes can approach that

Parallel scaling of the Maxwell solver – preliminary results



2D scattering from a cylinder

Fourth-order accurate



Fixed size problem, 3.8 million grid-points

Figure 1: Left: the computation of a shock hitting a cylinder (density). Right: parallel speedup for this problem, keeping the problem size fixed (4 Million grid points), on a linux cluster (Xeon processors).

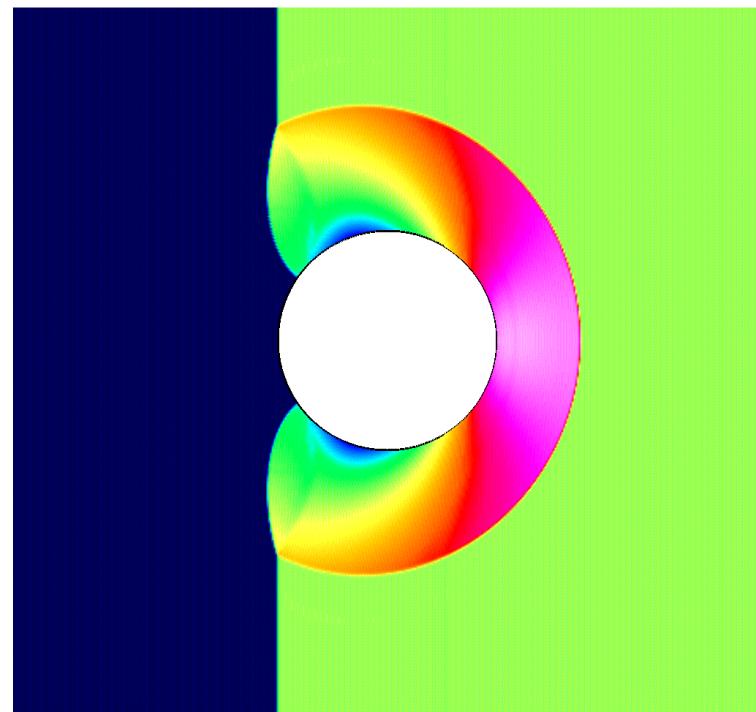
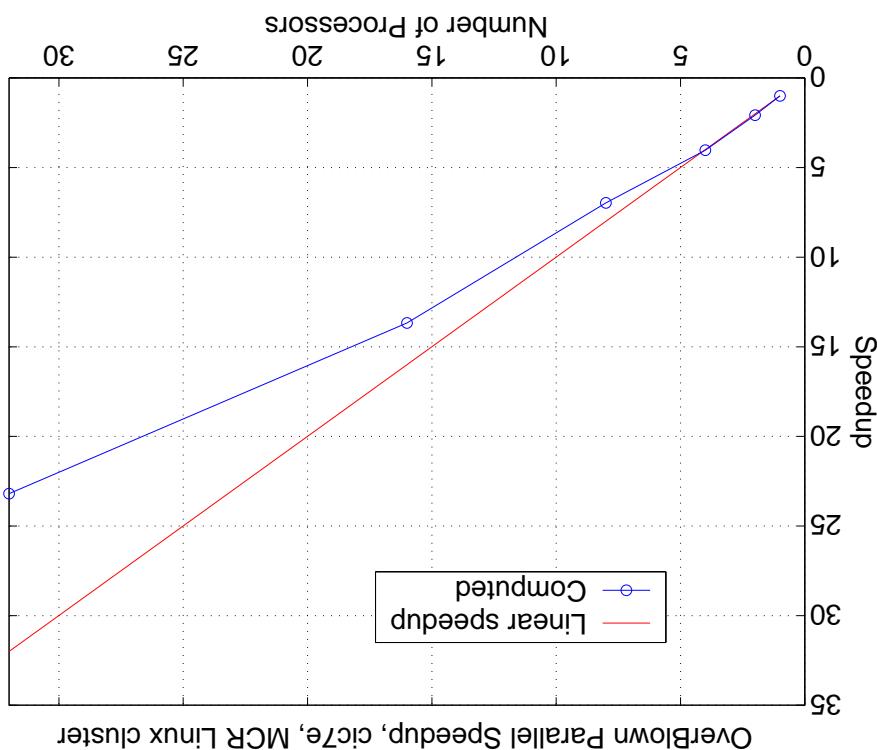
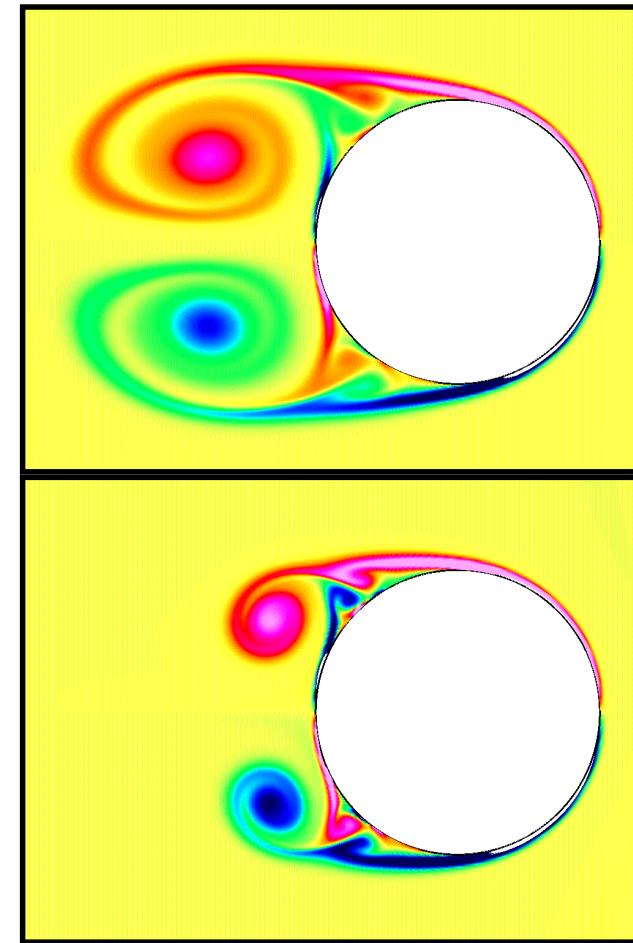
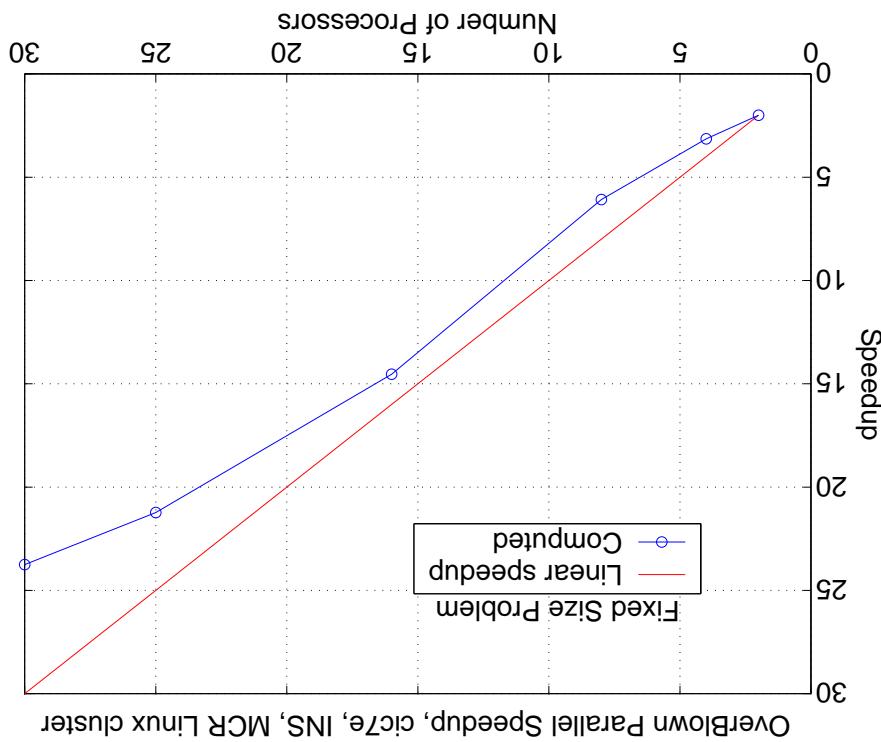


Figure 2: Left: impulsively started cylinder in an incompressible flow (vorticity). Right:
parallel speedup keeping the problem size fixed (4 Million grid points), on a linux cluster
(Xeon processors). The pressure equation is solved with algebraic multigrid (Hypre).



Incompressible Navier-Stokes: preliminary parallel results

- cylinders falling in a channel
- shock hitting multiple moving cylinders
- flow in a two-stroke engine

Show movies here...